



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS DE QUIXADÁ
CURSO DE ENGENHARIA DE SOFTWARE

FABIO JANYO FONTELES MAIA

AVALIAÇÃO DE ATIVIDADES DE TESTE POR USUÁRIOS FINAIS
NO CONTEXTO DO PROJETO SCALABLE GAME DESIGN BRASIL

QUIXADÁ
2018

FABIO JANYO FONTELES MAIA

**AVALIAÇÃO DE ATIVIDADES DE TESTE POR USUÁRIOS FINAIS
NO CONTEXTO DO PROJETO SCALABLE GAME DESIGN BRASIL**

Trabalho de Conclusão de Curso submetido à
Coordenação do Curso Bacharelado em
Engenharia de Software da Universidade Federal
do Ceará como requisito parcial para obtenção do
grau de Bacharel.

Área de concentração: Computação

Orientadora: Prof^a. Dra. Ingrid Teixeira Monteiro

QUIXADÁ

2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M186a Maia, Fabio Janyo Fonteles.
Avaliação de atividades de teste por usuários finais no contexto do projeto scalable game design Brasil /
Fabio Janyo Fonteles Maia. – 2018.
118 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá,
Curso de Engenharia de Software, Quixadá, 2018.
Orientação: Profa. Dra. Ingrid Teixeira Monteiro.

1. Desenvolvimento de usuário final. 2. Usuário final (Computadores). 3. Teste de software. 4.
Engenharia semiótica. 5. Design de jogos. I. Título.

CDD 005.1

FABIO JANYO FONTELES MAIA

AVALIAÇÃO DE ATIVIDADES DE TESTE POR USUÁRIOS FINAIS NO
CONTEXTO DO PROJETO SCALABLE GAME DESIGN BRASIL

Trabalho de Conclusão de Curso submetido à
Coordenação do Curso Bacharelado em Engenharia de
Software da Universidade Federal do Ceará como
requisito parcial para obtenção do grau de Bacharel.

Área de concentração: Computação

Aprovado em: ____/____/____

BANCA EXAMINADORA

Prof^ª. Dra. Ingrid Teixeira Monteiro (Orientadora)
Universidade Federal do Ceará – UFC

Prof^ª. Dra. Andréia Libório Sampaio
Universidade Federal do Ceará – UFC

Prof^ª. Dra. Paulyne Matthews Jucá
Universidade Federal do Ceará – UFC

Dedico a Deus e a minha família e amigos.

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus, por me permitir estar aqui e apresentar este trabalho, com saúde e determinação para conquistar meus objetivos.

Em segundo lugar, gostaria de agradecer a minha tia Lurdinha (*in memoriam*), que me criou como seu próprio filho, me dando educação de qualidade e estando ao meu lado em todos os momentos bons e ruins até seus últimos dias. Obrigado por acreditar no meu potencial e nunca desistir de mim. Obrigado por ter sido a minha família quando ninguém mais foi e que Deus te receba ao seu lado com todo seu amor e compaixão. Nunca irei esquecer-me da senhora e nem de seus ensinamentos.

Agradeço também ao meu pai, que me ajudou financeiramente para que eu pudesse estar em Quixadá e dar continuidade aos meus estudos. Sou muito grato ao meu irmão Junior, que muitas vezes me estendeu a mão quando minha tia não pôde, me ajudou a superar as dificuldades da vida e me ensinou a ver o mundo com mais empatia.

Assim como agradeço a estes parentes de sangue, agradeço também às minhas amigas, Beatriz e Cristina (*in memoriam*), pois, apesar de termos formados laços em diferentes etapas da vida, ambas me guiaram no melhor caminho possível e não me deixaram desistir, me levantaram quando eu caí e estavam ao meu lado para o que desse e viesse. Agradeço a minha orientadora Ingrid por toda a paciência que teve comigo, por me orientar da melhor forma possível, mesmo quando estava de licença.

“A grandeza da vida não consiste em não cair nunca, mas em nos levantarmos cada vez que caímos”.

Nelson Mandela

RESUMO

O desenvolvimento de aplicações por pessoas que não tem conhecimentos específicos sobre as atividades de engenharia de software, também conhecidos como usuário finais, tem, hoje, se tornado cada vez mais comum, e tornar esta experiência agradável e satisfatória é um dos principais objetivos da *End-user Software Engineering* (EUSE). Este trabalho apresenta um estudo de caso realizado no contexto do projeto *Scalable Game Design Brasil* (SGD-Br), que tem o objetivo de apoiar o aprendizado de raciocínio computacional por crianças e adolescentes, através da criação de jogos. A pesquisa traz uma abordagem sobre as atividades realizadas em um caso específico com uma das turmas participantes do projeto, focando na atividade de testes, que não fazia parte do roteiro do projeto, mas que foi vista como necessária pelo professor da turma, que também é um usuário final. Para concluir este objetivo, realizamos testes e inspeções nos jogos desenvolvidos pelos alunos, assim como uma análise nos resultados dos testes realizados por eles. Diante disso, pudemos verificar erros e conquistas obtidos pelos usuários finais, assim como conseguimos ver a importância de que abranger outras fases da EUSE pode impactar positivamente a qualidade do produto final desenvolvido por este tipo de usuário.

Palavras-chave: Usuário final. Testes Funcionais. MIS. EUSE. SGD-Br.

ABSTRACT

The development of applications by people who do not have specific knowledge about software engineering activities, also known as end-users, has become increasingly common today, and making this experience enjoyable and satisfying is one of End Users Software Engineering (EUSE) main goals. This paper presents a case study carried out in the context of the project Scalable Game Design Brazil (SGD-Br), which aims to support the learning of computational thinking by children and adolescents, through the creation of games. The research brings an approach to the activities carried out in a specific case with one of the classes participating in the project, focusing on the test activity, which was not part of the project script, but was seen as necessary by the class teacher, who is also an end user. To complete this goal, we conducted tests and inspections of games developed by the students, as well as an analysis of students' test results. On this, we were able to verify errors and achievements obtained by the end users, as well as we can see the importance of covering other phases of the EUSE can positively impact the quality of the final product developed by this type of user.

Keywords: End-User. Functional Testing. SIM. EUSE. SGD-Br.

LISTA DE FIGURAS

Figura 1 - Ciclo Geral da Pesquisa	23
Figura 2 - Ciclo de Execução do MIS	26
Figura 3 - Imagem do Frogger Original	30
Figura 4 - Imagem do Space Invaders Original.....	33
Figura 5 - Imagem do Sokoban Original	36
Figura 6 - Sokoban Desenvolvido pelo aluno A06.....	39
Figura 7 - Space Invaders Desenvolvido pelo aluno A02	40
Figura 8 - Sokoban Desenvolvido pelo Aluno A07	41
Figura 9 - Frogger Desenvolvido por A01	53
Figura 10 - Sokoban Desenvolvido por A01	55
Figura 11 - Space Invaders Desenvolvido por A01	56
Figura 12 - Frogger Desenvolvido por A02	57
Figura 13 - Space Invaders Desenvolvido por A02.....	59
Figura 14 - Space Invaders Desenvolvido por A02.....	60
Figura 15 - Sokoban Desenvolvido por A03	62
Figura 16 - Frogger Desenvolvido por A04	63
Figura 17 - Sokoban Desenvolvido por A04.....	64
Figura 18 - Space Invaders Desenvolvido por A04.....	66
Figura 19 - Frogger Desenvolvido por A05	67
Figura 20 - Sokoban Desenvolvido por A05	68
Figura 21 - Frogger Desenvolvido por A06	70
Figura 22 - Sokoban Desenvolvido por A06.....	71
Figura 23 - Frogger Desenvolvido por A07	73
Figura 24 - Sokoban Desenvolvido por A07.....	75
Figura 25 - Space Invaders Desenvolvido por A07.....	76
Figura 26 - Frogger Desenvolvido por A08	78
Figura 27 - Sokoban Desenvolvido por A08.....	79
Figura 28 - Space Invaders Desenvolvido por A08.....	80
Figura 29 - Sokoban Desenvolvido por A09	81
Figura 30 - Frogger Desenvolvido por A10	82
Figura 31 - Sokoban Desenvolvido por A10.....	83
Figura 32 - Space Invaders Desenvolvido por A10.....	85

Figura 33 - Space Invaders Desenvolvido por A11	87
Figura 34 - Sokoban Desenvolvido por A12	88
Figura 35 - Frogger Desenvolvido por A13	90

LISTA DE TABELAS

Tabela 1 - Relação entre alunos e jogos desenvolvidos	24
Tabela 2 - Roteiro Geral de Testes dos Jogos	42
Tabela 3 - Bugs constantes encontrados durante os testes funcionais.....	42

LISTA DE ABREVIATURAS E SIGLAS

IHC	Interação Humano-Computador
MIS	Método de Inspeção Semiótica
UFC	Universidade Federal do Ceará
EUSE	End-User Software Engineering
SGD	Scalable Game Design
SGD-Br	Scalable Game Design Brasil

SUMÁRIO

1	INTRODUÇÃO	15
2	TRABALHOS RELACIONADOS	17
2.1	Semiotic Inspection of a Game for Children with Down Syndrome	17
2.2	Signifying software engineering to computational thinking learners with AgentSheets and PoliFacets	18
3	FUNDAMENTAÇÃO TEÓRICA	19
3.1	End-User Development	19
3.2	EUSE – Testes	20
3.3	Scalable Game Design	21
3.4	Método de Inspeção Semiótica (MIS)	21
3.5	Testes Funcionais	22
4	PROCEDIMENTOS METODOLÓGICOS	22
4.1	Organizar Documentação e Informação Necessária	23
4.2	Execução e Análise do MIS	26
4.3	Realização de Testes Funcionais nos Jogos dos Alunos	27
4.4	Análise dos Testes dos Alunos	27
5	RESULTADOS	29
5.1	Execução e Análise do MIS	29
5.1.1	<i>Execução do MIS nos jogos originais</i>	29
5.1.1.1	<i>Frogger Original</i>	29
5.1.1.2	<i>Space Invaders Original</i>	32
5.1.1.3	<i>Sokoban Original</i>	35
5.1.2	<i>Execução e Análise do MIS dos jogos dos alunos</i>	38
5.2	Realização de Testes Funcionais nos Jogos dos Alunos	41
5.3	Análise dos Testes dos Alunos	43
5.4	Consolidação dos Resultados	45
6	CONSIDERAÇÕES FINAIS	49
7	TRABALHOS FUTUROS	50

REFERÊNCIAS	51
APÊNDICE A – Resultado dos MIS dos jogos dos alunos	53
APÊNDICE B – Análise dos testes dos alunos	92

1 INTRODUÇÃO

Diante do veloz crescimento da indústria computacional, principalmente de software, hoje, programação tornou-se uma técnica constante na vida de milhões de usuários. Com essa técnica em crescimento, no entanto, uma segunda tendência, provavelmente mais poderosa, tem tomado forma. De acordo com as pesquisas do US Bureau of Labor Statistic, em 2012, nos Estados Unidos, haveria menos de 3 milhões de programadores profissionais, para 55 milhões de pessoas que utilizavam planilhas e bases de dados no trabalho, muitas delas construindo fórmulas e consultas para apoiar suas tarefas (KO et al, 2011). Ou seja, o uso de programação tem se tornado quase tão difundido quanto o uso de computadores.

Uma das principais diferenças entre programadores profissionais e usuários finais são seus objetivos (KO et al, 2011). Programadores profissionais normalmente são pagos para criar e manter produtos de software, enquanto usuários finais escrevem programas para soluções pessoais de trabalhos ou *hobbies*. Apesar das diferenças nas prioridades de desenvolvedores profissionais, usuários finais enfrentam muitos dos mesmos desafios de engenharia de software, como por exemplo, definição das funcionalidades, que corresponde à etapa de elicitação de requisitos em linguagem técnica, escolha de ferramentas para organização do ambiente de programação, testes para verificar e eliminar erros de seus projetos, entre outros.

Além dos objetivos, *end-users* e programadores apresentam diferenças também no âmbito de desenvolvimento. Programadores profissionais tendem a ter uma série de preocupações com fatores como confiabilidade, reutilização e manutenção (KO et al, 2011). Eles não têm apenas a intenção de criar algo, e sim a intenção de desenvolver um produto robusto, softwares que irão atender às necessidades dos usuários de modo eficaz, competente e duradouro, ou seja, a principal diferença entre programação profissional e programação pelo usuário final é a atenção dada aos resultados do trabalho e à qualidade do produto de software. Contudo, o usuário final também se preocupa com a qualidade de seus produtos, porém, de forma secundária (KO et al, 2011).

Nesta pesquisa, será analisado um caso excepcional ocorrido em um projeto de pesquisa chamado Scalable Game Design Brasil (SGD-Br), onde, resumidamente, usuários finais, neste caso, alunos de ensino fundamental, testaram os jogos desenvolvidos pelos seus companheiros de turma e registraram os resultados em uma planilha criada pelo professor.

O aspecto particular deste contexto é que ambos, alunos e professor, não possuíam conhecimentos técnicos sobre engenharia de software, como métodos de testes e avaliações de softwares.

O projeto SGD-Br tem o objetivo de apoiar o aprendizado de raciocínio computacional por crianças e adolescentes, através da criação de jogos. O projeto adotou como ferramenta de desenvolvimento a plataforma AgentSheets, a partir da qual usuários podem construir jogos, simulações, demonstrações interativas, entre outros tipos de programas. E, como plataforma de publicação, o projeto utiliza o PoliFacets, ambiente sociotécnico desenvolvido para apoiar o ensino de raciocínio computacional no contexto do projeto Scalable Game Design Brasil (POLIFACETS, 2012). A exploração das “facetas” disponibilizadas no sistema permite explicitar conceitos do AgentSheets e entender cada detalhe dos projetos criados pelos alunos.

O projeto contou com a parceria de três escolas localizadas nas cidades do Rio de Janeiro e Niterói, onde a pesquisa foi executada. Em cada escola, os professores integrantes do projeto davam instruções aos alunos e estes criavam seus jogos na plataforma do AgentSheets, subindo suas aplicações para o ambiente PoliFacets, onde os alunos deveriam escrever a descrição e o objetivo do jogo.

Na escola EA, uma escola internacional com aulas ministradas em inglês, onde ocorreu o caso estudado, foi selecionada uma turma de 30 alunos subdividida em 2 grupos: TA, com dezoito alunos e TB, com doze alunos. Nas duas turmas, o professor pediu para que os alunos tentassem recriar jogos clássicos disponíveis na biblioteca de projetos do AgentSheets. Eles tinham que desenvolver suas próprias versões dos jogos no AgentSheets e enviá-los para a plataforma PoliFacets. Vendo que alguns alunos estavam enviando jogos com alguns problemas, o professor responsável pela turma decidiu, por conta própria, que ali seria executada uma atividade de testes, onde os alunos teriam de testar e avaliar os jogos dos colegas respondendo a algumas perguntas de um questionário criado pelo professor.¹

Neste trabalho, foram analisadas as atividades relacionadas aos testes de software que ocorreram durante a execução do caso abordado. Foram avaliados o perfil da turma, os jogos desenvolvidos pelos participantes, os problemas encontrados nestes jogos e a qualidade dos testes realizados pelos alunos.

¹ As informações apresentadas sobre o projeto, a turma e a escola foram obtidas a partir de relatos de membros do projeto e a partir da documentação disponibilizada (anotações de pesquisadores, transcrições de entrevistas e etc) e, por razões éticas, foram mantidas sob anonimato.

O texto é dividido em oito seções, iniciando com os trabalhos relacionados, que são pesquisas de terceiros que, de alguma forma, nos ajudaram a obter clareza sobre assuntos abordados nesta pesquisa. Logo após, temos a fundamentação teórica, onde esclarecemos termos, teorias e metodologias utilizadas nesta pesquisa. Seguindo com os procedimentos metodológicos, mostrando o passo a passo da execução da pesquisa e, em seguida, os resultados dos passos tomados. Ademais temos a seção de considerações finais do trabalho, uma seção para trabalhos futuros e as referências desta pesquisa. Por último temos os apêndices, que mostram resultados das metodologias utilizadas.

2 TRABALHOS RELACIONADOS

Esta seção traz dois trabalhos que, de alguma forma, nos ajudaram a entender melhor determinados assuntos abordados na pesquisa. O primeiro traz exemplos do uso do método de inspeção semiótica, método proposto pela engenharia semiótica, uma teoria da Interação Humano-Computador (IHC) que busca investigar a comunicação entre usuários, sistemas e designers. *“Os processos de comunicação investigados são realizados em dois níveis distintos: a comunicação direta usuário-sistema e a metacomunicação do designer para o usuário medida pelo sistema, através de sua interface”* (BARBOSA, SILVA, 2010, p. 77). Já o segundo trabalho mostra a importância da EUSE e das tecnologias escolhidas pelos usuários e como estas podem impactar no produto final desenvolvido pelo usuário. Além disso, sua pesquisa traz uma análise sobre o AgentSheets e o PoliFacets, que são as mesmas tecnologias utilizadas pela turmas do SGD-Br sobre o qual nossa pesquisa aborda.

2.1 Semiotic Inspection of a Game for Children with Down Syndrome

O trabalho de Brandão et al. (2010) traz uma abordagem sobre análise de jogos para crianças com síndrome de Down. Segundo os autores, não existem muitas iniciativas na área de jogos para crianças especiais, principalmente para crianças com esse tipo de mutação genética.

A pesquisa parte da análise do jogo JECRIPE, especialmente criado pela equipe de desenvolvimento para crianças com síndrome de Down, usando o método de inspeção semiótica (MIS) para estudar e entender a interação do usuário e então examinar seus aspectos positivos e os problemas de comunicabilidade encontrados.

Baseando-se neste trabalho, a realização do MIS nos jogos feitos pelos participantes do SGD-Br torna-se mais fácil, já que os dois trabalhos apoiam a realização da inspeção em jogos, mesmo que em contextos diferentes. No entanto, o MIS não se diferencia por contexto, e os passos a serem seguidos são basicamente os mesmos.

2.2 Signifying software engineering to computational thinking learners with AgentSheets and PoliFacets

Programas não profissionais de usuários finais aumentaram notavelmente em volume e diversidade. No entanto, para que esses programas sejam utilizáveis e confiáveis, seus criadores devem estar familiarizados com práticas de engenharia de software que normalmente não fazem parte de sua faixa de competência (Monteiro et al, 2017).

A pesquisa de Monteiro et al apresenta a importância da aquisição de pensamento computacional (CTA), mas, além disso, aborda a importância da aquisição de noções de qualidade de software, apresentando um relato sobre um programa de aquisição de pensamento computacional brasileiro guiado por princípios semióticos e apresenta um estudo sobre como as tecnologias utilizadas durante o desenvolvimento de software impactam na qualidade do produto final.

Apesar das diferenças entre os objetivos entre nossa pesquisa e a pesquisa de Monteiro et al., ambas trazem uma análise do projeto SGD-Br focando atividades de engenharia de software pelo usuário final (EUSE), mostrando que diversos de seus elementos estão presentes, mesmo que de forma implícita, nas atividades do usuário final. Além disso, esta pesquisa mostra a importância e o impacto que a EUSE e as tecnologias escolhidas trazem para o produto final desenvolvido pelo usuário.

3 FUNDAMENTAÇÃO TEÓRICA

Nesta revisão bibliográfica, serão abordadas as concepções utilizadas na execução da nossa pesquisa. Na primeira subseção, será definido o conceito de *End-user Development* (EUD) e falaremos um pouco sobre sua diferença em relação ao desenvolvimento de software por pessoas mais experientes na área. Na segunda subseção, focaremos na atividade de teste no contexto de EUSE. Na terceira subseção falaremos sobre o Scalable Game Design e sobre o Scalable Game Design Brasil, projetos que possibilitaram a realização desta pesquisa. Na quarta, será explanado o método de avaliação utilizado para analisar e conhecer os jogos desenvolvidos pelas crianças: o Método de Inspeção Semiótica (MIS). Na quinta e última subseção, definiremos o que são testes funcionais e quais são os resultados esperados ao realizá-los.

3.1 End-User Development

End-user development é um conjunto de técnicas, métodos e ferramentas que permitem que os usuários de sistemas de software, que atuam como desenvolvedores de software não profissionais, em algum momento criem, modifiquem ou estendam um artefato de software (LIEBERMAN et al, 2006, apud BURNETT et al, 2014). De modo informal, o EUD permite aos usuários finais projetar ou personalizar a interface do usuário e as funcionalidades do software, assim como a programação técnica (BURNETT, 2014).

Uma das principais vantagens do EUD é que o usuário final conhece quais são as suas necessidades e o contexto do que precisa ser criado ou modificado. A EUD permite a estes usuários finais sintonizar seus softwares para atender estas necessidades da maneira mais próxima possível do que seria sem o EUD.

Segundo Burnett (2014), EUD é inerentemente diferente de desenvolvimento de software tradicional, e tentar apoiar o EUD simplesmente imitando as abordagens tradicionais é muitas vezes insuficiente para produzir resultados bem-sucedidos. Isso ocorre porque geralmente estes usuários finais não têm conhecimento de linguagens de programação, processos formais ou modelagem de diagramação. Além disso, este tipo de usuário geralmente não tem tempo ou motivação para aprender estes tipos de técnicas formais, já que, de acordo com Burnett (2014), usuários finais normalmente escrevem código a fim de

alcançar um objetivo a curto ou médio prazo, ao invés de criar um software durável. De qualquer forma, apoiar o EUD requer fornecer ferramentas adequadas, estruturas sociais e processos de desenvolvimento que são altamente usáveis, de fácil aprendizado e facilmente integrados em prática de domínio (BURNETT, 2014).

Na pesquisa, foram analisados artefatos encontrados em ambientes criados para utilização pelos alunos, por exemplo, o PoliFacets, onde é feito o upload dos jogos, além de técnicas utilizadas por eles para criação dos jogos e para realização da avaliação da atividade de teste. Além disso, os jogos dos alunos foram criados na plataforma AgentSheets, que também é uma ferramenta destinada para o desenvolvimento de aplicações do usuário final.

3.2 EUSE – Testes

Atualmente, existem várias formas de testar se programas estão funcionando corretamente, todas elas têm um objetivo em comum, que é verificar se o produto está funcionando da maneira correta de acordo com os requisitos do sistema. De acordo com Ko et al (2011), as técnicas de teste e verificação tem um objetivo universal: elas permitem que as pessoas tenham um nível maior de confiança e precisão do que elas teriam se pulassem essa etapa, e o que difere um profissional de um usuário final é o nível de confiança que eles têm sobre a corretude de seus programas. De acordo com seu trabalho, a pesquisa voltada para testes e verificação de sistemas EUD tem o foco em ajudar os usuários finais a gerenciarem seu excesso de confiança e, principalmente, o paradigma do uso de planilhas², que tendem a gerar resultados mesmo na presença de erros.

A pesquisa de Ko et al (2011) mostra diversas metodologias e abordagens que são utilizadas por *end-users* ou que podem ajudar o usuário final a superar estes problemas de confiança. Como um dos resultados desta pesquisa, a partir da análise dos testes dos alunos, que também são usuários finais, foi possível observar em qual destas metodologias os testes dos alunos se encaixavam para aprimorar o nível da análise dos testes.

² Uma possível explicação para esse viés é que o Microsoft Excel, uma das linguagens de planilhas mais utilizadas, tende a produzir saídas mesmo na presença de erros. Isto leva o usuário a criar um excesso de confiança sob a corretude de seus programas, o que pesquisadores tem tentado resolver com melhores ferramentas de testes.

3.3 Scalable Game Design

Iniciativa liderada pelo professor Alexander Repenning na universidade de Colorado, em Boulder – EUA, o Scalable Game Design é um projeto que tem como missão reinventar a informática em escolas públicas, motivando e educando todos os alunos, incluindo mulheres e comunidades sub-representadas para aprender sobre ciência da computação através de construção de jogos começando no nível do ensino médio (SCALABLE GAME DESIGN, 2009).

O projeto Scalable Game Design Brasil é associado ao projeto Scalable Game Design, nos EUA. Enquanto o projeto americano tem como objetivo inovar o ensino de computação em escolas públicas dos Estados Unidos, no Brasil, o projeto Scalable Game Design Brasil ou “SGD-Br”, visa desenvolver uma tecnologia brasileira para facilitar o ensino do raciocínio computacional em escolas de ensino fundamental e médio, sejam elas públicas ou particulares (SCALABLE GAME DESIGN BRASIL, 2011).

3.4 Método de Inspeção Semiótica (MIS)

O Método de Inspeção Semiótica (MIS) proposto pela engenharia semiótica, examina uma grande diversidade de signos os quais o usuário é exposto à medida que este interage com artefatos de computação (DE SOUZA, 2006). Seu foco está na emissão da mensagem de metacomunicação do designer e não envolve o usuário (SALGADO, 2007). O inspetor, que deve ser um especialista em IHC, busca reconstruir a mensagem a partir da avaliação dos signos do sistema.

No projeto, os jogos desenvolvidos pelos alunos da turma selecionada passaram por uma inspeção utilizando o MIS onde foram identificados os objetivos apoiados pelo sistema, inspecionadas as interfaces e analisando os signos estáticos, dinâmicos e metalinguísticos para a reconstrução da metamensagem correspondente.

O objetivo da aplicação do MIS nesta pesquisa é obter lucidez sobre os jogos desenvolvidos pelos participantes do projeto e a mensagem que os designers tentaram transmitir através daquele sistema. A partir daí, foi possível realizar a prática de engenharia reversa dos jogos, mesmo sem analisarmos algum tipo de documentação sobre o desenvolvimento do jogo, pois estes não foram elaborados nem pela equipe de projeto e nem pelos alunos participantes do projeto.

3.5 Testes Funcionais

Softwares são desenvolvidos por seres humanos, que por mais capazes e experientes que sejam, não são perfeitos, sendo passíveis de cometerem erros e, por isso, é comum encontrar erros em softwares em desenvolvimento e até mesmo completos. Mesmo que a equipe de desenvolvimento não encontre estes erros, é provável que os usuários daquele sistema o encontrem, já que cada vez que ele executa determinada ação naquele sistema, ele está exercitando uma funcionalidade.

De acordo com Coutinho (2011), os testes funcionais também são conhecidos como "testes caixa-preta", pelo fato de o testador não precisar conhecer os detalhes da codificação. Nesse tipo de teste, o testador informa os dados de entrada e verifica se a saída/resultado está de acordo com o que era esperado. Portanto, esse tipo de teste não tem como objetivo saber como a funcionalidade foi implementada, mas sim quais são os resultados apresentados.

Avaliando somente a entrada e a saída, o teste de caixa-preta visa identificar defeitos do tipo:

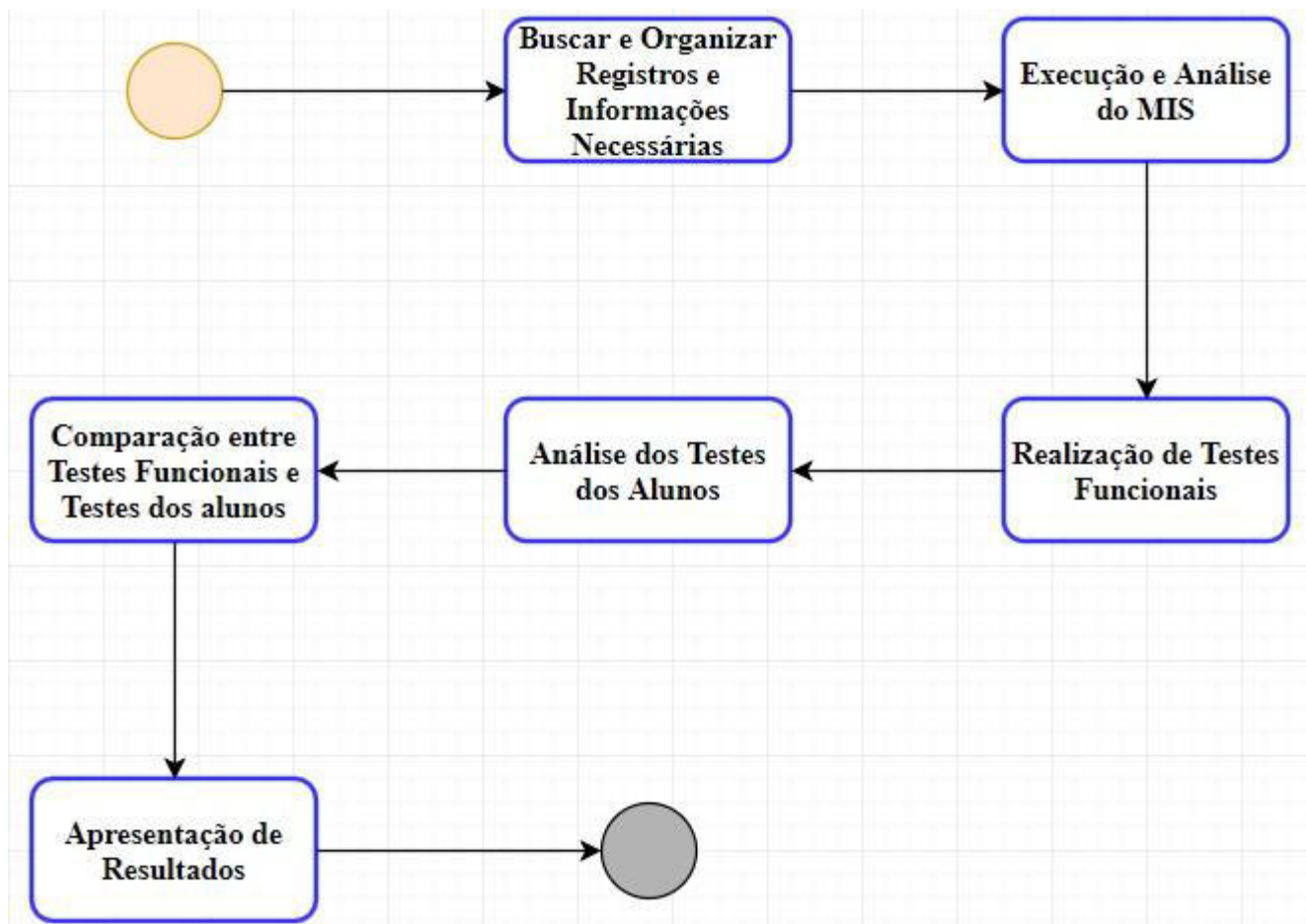
- Se o sistema aceita entradas incorretas;
- Se a saída produzida está correta;
- Se existem erros na interface;
- Se alguma funcionalidade está faltando;

Para realizar uma melhor análise dos testes dos alunos, realizamos testes funcionais nos jogos desenvolvidos por eles para que pudéssemos comparar os resultados e fazer um levantamento sobre as principais diferenças na realização de testes pelo usuário final e por uma pessoa com mais conhecimento na área.

4 PROCEDIMENTOS METODOLÓGICOS

Nesta seção será listado e explanado cada passo, separadamente, dos procedimentos metodológicos, como mostra a Figura 1.

Figura 1. Ciclo Geral da Pesquisa



Fonte: Elaborado pelo autor

4.1 Organizar Documentação e Informação Necessária

Nossa pesquisa parte de um estudo de caso do projeto Scalable Game Design Brasil que já foi finalizado e, para poder analisá-lo, foi necessário buscar fontes de informações para dar continuidade a nossa pesquisa.

Primeiramente, buscamos os jogos dos alunos da turma selecionada, que se encontravam no perfil de cada aluno no PoliFacets. Embora fosse possível utilizar um perfil de administrador do sistema para ter acesso a todos os jogos, decidimos não utilizar este perfil, pois este continha permissões que poderiam comprometer informações privadas do projeto, como nome do aluno e da escola. Por isso, foi criado um novo perfil no PoliFacets especialmente para que pudéssemos subir os jogos anonimizados e realizar o que fosse necessário com eles. No entanto, alguns dos jogos possuíam problemas no dimensionamento da área dos jogos, não permitindo que pudéssemos realizar qualquer análise, pois não era possível executar e jogar normalmente. Esse problema estava relacionado com o

desenvolvimento dos jogos por parte dos alunos, pois durante a execução do projeto SGD-Br, os jogos foram testados dessa maneira, com problemas de tamanho com a planilha (*worksheet*)³, que é como chamamos a área de trabalho do AgentSheets, mas mesmo assim tomamos a decisão de consertá-los para que pudéssemos realizar o MIS e os testes funcionais. No entanto, estes consertos não influenciaram nos resultados dos testes funcionais, pois os resultados dos MIS foram utilizados apenas para identificar quais eram os objetivos apoiados pelos sistemas, no caso, os jogos.

A segunda fonte de informações foi um arquivo em Excel disponibilizado por um dos participantes do projeto SGD-Br que acompanhou a turma e o professor durante as atividades realizadas no caso específico da análise desta pesquisa. Este material continha uma lista com informações sobre todos os alunos da turma selecionada para análise, quais jogos haviam sido desenvolvidos pelos alunos, como pode ser visto na Tabela 1, nome dos jogos, descrições e instruções dos jogos (escritas pelos alunos), uma tabela mostrando quem testou o jogo de quem e uma informação crucial, quais alunos entregaram ou não a permissão assinada pelos pais ou responsáveis para que o projeto pudesse utilizar o que teria sido feito pelo aluno na execução do projeto para pesquisa. Infelizmente, quatro dos alunos da turma selecionada não entregaram este documento assinado, impossibilitando o uso de qualquer dado destes alunos para pesquisa.

Tabela 1. Relação entre alunos e jogos desenvolvidos

Aluno	Frogger	Space Invaders	Sokoban
A01	X	X	X
A02	X	X	X
A03	—	—	X
A04	X	X	X
A05	X	—	X
A06	X	—	X
A07	X	X	X
A08	X	X	X

³ No decorrer do texto, as planilhas (*worksheets*) dos jogos serão referidas como “mapa de *tiles*”, que é uma expressão mais comum para a área de trabalho dos jogos em geral.

A09	—	—	X
A10	X	X	X
A11	—	X	—
A12	—	—	X
A13	X	—	—
A14	—	—	—

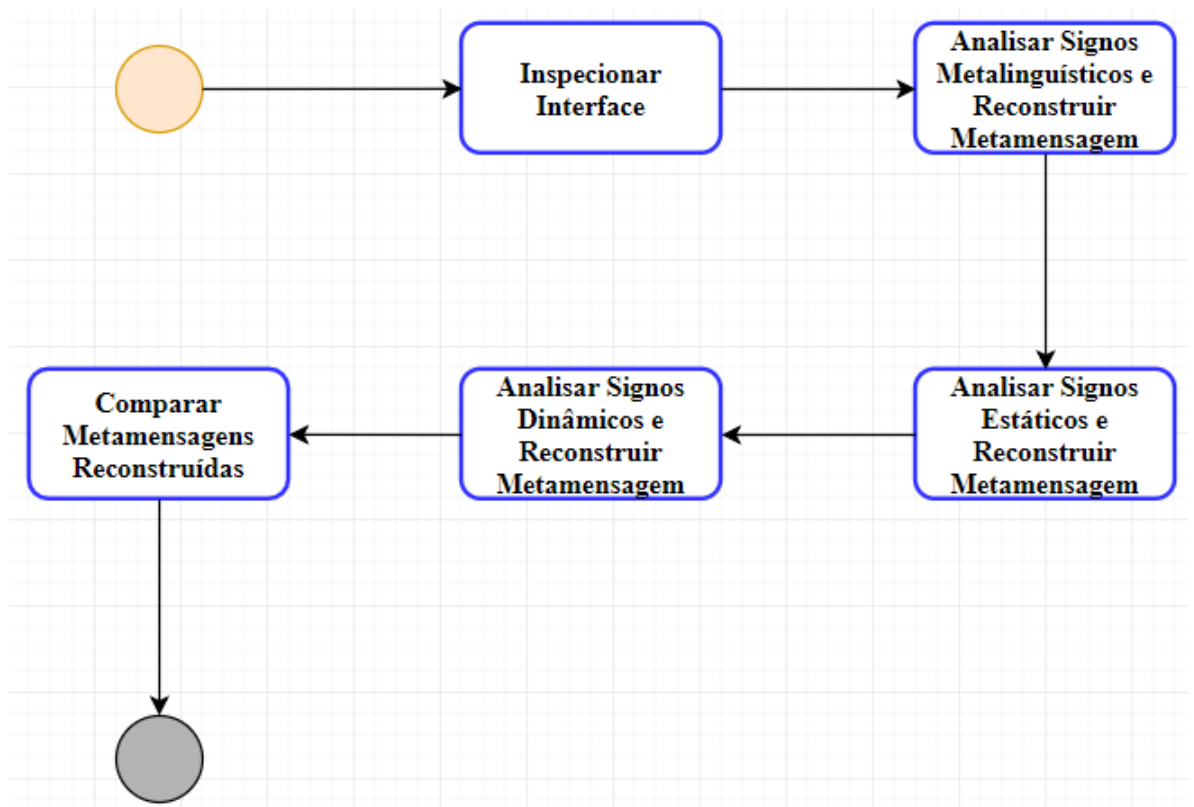
Fonte: Elaborada pelo autor

Sobre os detalhes da execução do projeto, tivemos a página web do SGD-Br, que dispôs diversas informações gerais sobre o SDG-Br e informações detalhadas por meio de relatos de um dos participantes do projeto, que acompanhou a turma durante as aulas. No entanto, como não tivemos acesso a gravações ou registros da época do projeto, tivemos algumas dificuldades com estes relatos, pois o projeto aconteceu há mais de um ano e o participante não se lembrava de todos os detalhes, então trabalhamos com o máximo de informações possível que ele foi capaz de recordar.

4.2 Execução e Análise do MIS

Como citado anteriormente, o Método de Inspeção Semiótica (MIS) é um método proposto pela Engenharia Semiótica, cujo foco está na emissão da mensagem de metacomunicação do designer e não envolve o usuário (SALGADO, 2007). O inspetor, que deve ser um especialista em IHC, busca reconstruir a mensagem a partir da análise da interface e da avaliação dos signos do sistema, como mostra a Figura 2.

Figura 2. Ciclo de Execução do MIS



Fonte: Elaborado pelo autor

De acordo com Barbosa e Silva (2010), os signos são divididos em três tipos: estáticos, dinâmicos e metalinguísticos. Na visão de Salgado (2007), signos estáticos são aqueles encontrados nas telas estáticas do sistema, devem ser interpretados independente de interações ou relações temporais com o sistema. São exemplos de signos estáticos: layout geral e a disposição de elementos em uma tela, itens de menu, botões de uma barra de ferramentas, campos e botões de um formulário e o conteúdo expresso em um texto, lista, tabela, árvore ou outras formas de visualização que não incluem animações (BARBOSA E SILVA, 2010). Signos dinâmicos são aqueles que expressam o comportamento do sistema e

são acessíveis somente quando o usuário interage com o artefato (SALGADO, 2007). Por estarem vinculados à interação, esses tipos de signos devem estar sempre fazendo referência a ela. São exemplos de signos dinâmicos: a associação causal entre a escolha de um item de menu e a exibição do diálogo, a possibilidade de arrastar os itens de uma área da tela para outra e o surgimento de uma dica sobre um elemento de interface ao ser sobreposto pelo cursor do mouse. Salgado (2007) diz que os signos metalinguísticos são aqueles usados pelo designer para comunicar explicitamente para os usuários os significados que ele atribuiu aos demais signos codificados na interface e como eles devem ser usados. Em geral, ocorrem na forma de mensagens de ajuda e de erro, alertas, diálogos de esclarecimento, dicas e assemelhados (BARBOSA e SILVA, 2010). O objetivo da análise de signos metalinguísticos é identificar elementos importantes usados no discurso dos designers. Os signos metalinguísticos são usados para fornecer uma melhor compreensão dos signos estáticos e dinâmicos, bem como a relação entre eles (BRANDÃO et al. 2010). A classificação dos signos orienta o trabalho do avaliador durante a inspeção semiótica.

4.3 Realização de Testes Funcionais nos Jogos dos Alunos

A realização do MIS levou a um entendimento maior sobre os jogos, e o conhecimento sobre as metamensagens nos permitiu ter uma visão sobre os objetivos apoiados pelo sistema, ou seja, a partir do conhecimento dos objetivos do jogo, dos seus obstáculos, regras e mensagens de aviso e erros, ficou mais fácil perceber as funcionalidades presentes nos jogos. Após isso, estávamos aptos a realizar testes funcionais ou testes de caixa preta.

Testes funcionais, como já dito anteriormente, não necessitam de nenhuma inspeção de código, apenas fornecemos entradas específicas no sistema (comandos do jogo) e analisamos como este reagia (comportamentos no jogo), analisando se a saída apresentada por ele estava de acordo com o que o desenvolvedor tinha intenção de criar, conhecida a partir da inspeção com MIS.

4.4 Análise dos Testes dos Alunos

Esta etapa se deu a partir da análise dos resultados dos 46 testes dos alunos, que se encontravam registrados na planilha fornecida pelo participante do projeto, contendo quarenta e seis resultados de testes. Essa planilha mostrava as respostas dos alunos em relação aos

testes executados por ele sobre os jogos de seus colegas, contendo os seguintes campos (definidos previamente pelo professor da turma):

- *Who is the developer of this game?* (Quem é o desenvolvedor deste jogo?)
- *What is the title of this game?* (Qual o título do jogo?)
- *Is the description and instructions for this game well explained and sufficient?*
(A descrição e as instruções deste jogo são suficientes e estão bem explicadas?)
- *Does this game have any bug?* (Este jogo contém algum bug?)
- *What and where are the bugs? How do you recommend fixing them?* (Quais e onde estão os bugs? Como você recomenda que eles sejam consertados?)
- *Do you have any suggestion for improvement? What are those suggestions?*
(Você tem alguma sugestão para melhoria? Quais são essas sugestões?)

A partir da investigação desta planilha, pudemos realizar análises qualitativas e quantitativas da atividade de teste no âmbito do usuário final, investigando as práticas utilizadas pelos alunos, tipos de bugs que foram ou não encontrados, além de realizar uma comparação entre os bugs encontrados pelos alunos e os bugs encontrados durante a execução de testes funcionais por nós.

5 RESULTADOS

Nesta seção, são apresentados os resultados da pesquisa, divididos entre os resultados do MIS, os resultados da execução dos testes funcionais e as análises sobre os testes dos alunos.

5.1 Execução e Análise do MIS

Nesta subseção serão apresentados os resultados do MIS executado nos jogos originais, aqueles em que os alunos basearam seus projetos, e nos jogos desenvolvidos pelos alunos da turma.

Durante a execução do método de inspeção semiótica, o avaliador realizou a inspeção dividida por jogos, onde cada jogo teve a inspeção completa, os resultados foram organizados de acordo com a classificação dos signos, em Metalinguísticos, Estáticos e Dinâmicos. Os signos foram destacados e interpretados, e, após isso, a metamensagem de cada classe de signos foi apresentada, de acordo com o entendimento do avaliador, sobre a intenção do designer ao desenvolver essa interface.

5.1.1 Execução do MIS nos jogos originais

Os primeiros jogos escolhidos para iniciar o MIS foram os jogos encontrados na seção de projetos de exemplo na plataforma do PoliFacets, que contém uma lista de diversos jogos que vinham como projetos de exemplo no AgentSheets. O professor da turma pediu para que os alunos desenvolvessem seus jogos baseando-se em três jogos específicos desta lista: Frogger, Space Invaders e Sokoban, que são reproduções de jogos clássicos do console Atari.

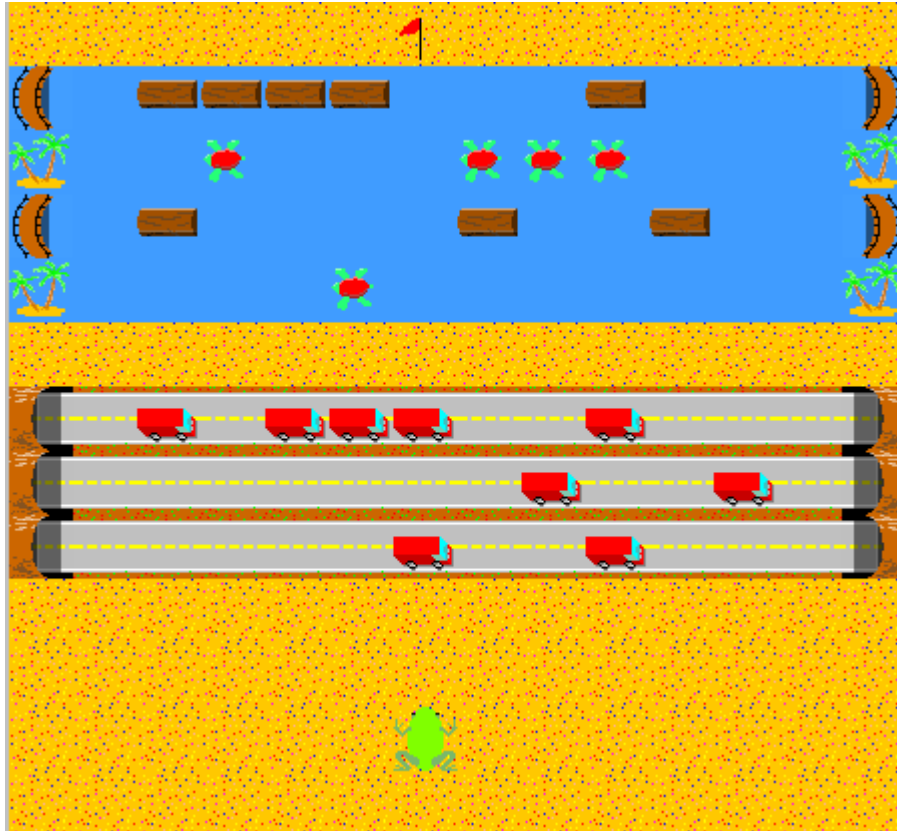
O objetivo da análise dos jogos originais é apenas conhecer melhor os tipos de jogos que foram desenvolvidos pelos alunos. A partir daí, podemos aprimorar a análise dos jogos desenvolvidos pelos alunos e saber o que os diferencia dos jogos que eles tomaram como base, assim como o que diferencia o jogo de um aluno para o jogo do outro.

5.1.1.1 Frogger Original

O “Frogger” foi um dos três jogos escolhidos para que os alunos recriassem durante a execução do SGD-Br. Nele, nós temos o personagem principal (sapo) com o objetivo de

atravessar o rio, passando pelos obstáculos (caminhões, rio), para chegar ao objetivo final (bandeira), como pode ser observado na figura 3.

Figura 3. Imagem do Frogger Original



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- **Descrição do jogo** (*Ajude o sapo a evitar os obstáculos para chegar até ao outro lado do rio.*): Fala sobre o objetivo do jogo e que existe um problema que deve ser superado para atingi-lo.
- **Instruções** (*Use as setas para mover o sapo.*): Fala objetivamente sobre os controles do jogo.
- **Mensagem ao cair na água** (*I cannot swim*): Informa que o sapo não consegue nadar e, por isso, o jogador perdeu.
- **Mensagem ao tentar passar pelos túneis laterais** (*Cannot walk over tunnels. That's cheating!*): Informa que, ao tentar passar por este objeto, o jogador estaria tentando trapacear e, por isso, perdeu.

- **Mensagem ao tentar passar pelas ilhas laterais** (*Cannot walk over a turtle maker. That's cheating!*): Informa que, ao tentar passar por este objeto, o jogador estaria tentando trapacear e, por isso, perdeu.
- **Mensagem ao chegar à bandeira** (*I made it!*): Informa que o jogador conseguiu cumprir o objetivo e finalizou o jogo.

Metamensagem

Na visão do designer, o usuário é alguém que gosta de jogos simples e objetivos. Como é comum em muitos jogos, neste também há ações proibidas e situações de perigos presentes. O jogo tem seu uso facilitado, através de comandos simples como as setas do teclado. Ele também se baseia em um usuário que gosta de receber feedbacks quando ocorre algo inesperado no jogo como morte do personagem principal ou andar por locais proibidos. As mensagens são passadas em primeira pessoa, o que passa a ideia de independência do personagem, destacando o contexto em que o jogo está inserido.

SIGNOS ESTÁTICOS:

- **Sapo:** Representa o personagem principal do jogo
- **Gramma:** Representa uma área segura para o sapo
- **Estrada:** Representa uma área de perigo para o sapo
- **Bandeira:** Representa o ponto de conclusão da fase
- **Túnel:** Representam os pontos de onde os caminhões aparecem e somem do jogo
- **Ilha:** Representam os pontos de onde as tartarugas aparecem e somem do jogo
- **Ponte:** Representam os pontos de onde os troncos aparecem e somem do jogo
- **Água:** Representa uma área de perigo para o sapo
- **Tronco:** Representa objeto que ajuda o sapo a passar pela água sem morrer
- **Tartaruga:** Representa objeto que ajuda o sapo a passar pela água sem morrer
- **Caminhão:** Representa um obstáculo que pode matar o sapo

Metamensagem:

O designer utiliza elementos que são conhecidos da rotina do usuário, como automóveis, animais, pontes, etc., o que torna o ambiente do jogo mais intuitivo, pois segue o mesmo padrão da realidade. O usuário, provavelmente uma criança, prefere cenários

coloridos e gosta de um tempo para analisar o jogo antes de entrar em zonas perigosas, pois o sapo, que é o personagem principal, começa na grama, que é uma larga área segura para ele.

SIGNOS DINÂMICOS:

- **Sapo:** Com a ação de movimento do jogador, é o personagem principal e controlado do jogo
- **Tronco:** O movimento do tronco representa ação causal que pode matar o sapo no rio
- **Tartaruga:** O movimento da tartaruga representa ação causal que pode matar o sapo no rio
- **Caminhão:** O movimento do caminhão representa ação causal que pode matar o sapo na rodovia

Metamensagem:

O designer se baseia em um usuário que gosta de ter livre movimentação e tempo para pensar, pois o sapo pode andar para trás e voltar nos obstáculos. Além disso, o jogo apresenta diferentes perigos ao sapo, representados pelas ações dos caminhões e o risco de o sapo não conseguir subir nos troncos e tartarugas, que se movimentam para que não seja simples a travessia do sapo pelo rio, o que torna perceptível que o usuário gosta de desafios para poder chegar ao objetivo.

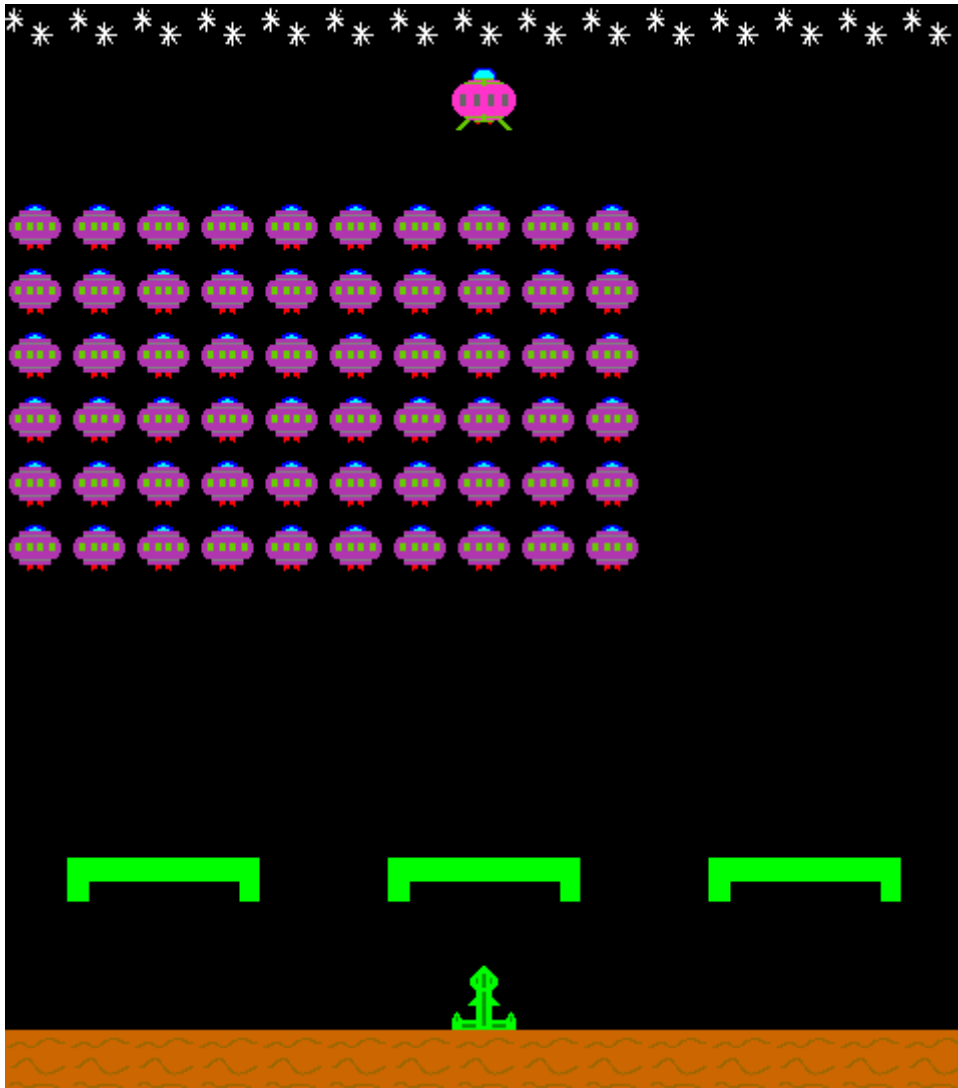
COMPARAÇÃO

Nos três tipos de signos os elementos do jogo são apresentados com o mesmo sentido, sem nenhuma divergência percebida. Além disso, apesar de o designer não deixar claro que o usuário é uma criança, ele faz uso de signos estáticos e dinâmicos que facilitam o entendimento do jogo, já que ele usa elementos vistos no dia-a-dia de uma criança e não altera o sentido deles, além de ter uma movimentação fácil e um objetivo simples.

5.1.1.2 Space Invaders Original

O segundo jogo escolhido para ser recriado pelos alunos foi o Space Invaders. Nele, o personagem principal se encontra como um canhão, que tem o objetivo de defender a Terra dos alienígenas invasores, como pode ser observado na Figura 4.

Figura 4. Imagem do Space Invaders Original



Fonte: PoliFacets

SIGNOS METALINGUISTICOS:

- **Descrição do jogo** (*Sua missão é destruir todas as naves inimigas*): Deixa claro o objetivo do jogo, apresentando um pouco do contexto no qual o jogo está inserido.
- **Instruções** (*Mova o canhão usando as setas e use a barra de espaço para atirar.*): Mostra, objetivamente, os controles do jogo.
- **Mensagem ao ser atingido por um tiro inimigo** (*Your cannon was destroyed! Game Over!*): Mensagem informando que o jogador foi derrotado e o jogo chegou ao fim.

Metamensagem:

O designer assume que o usuário é alguém que gosta de jogos simples e objetivos, que não necessita de uma história, apenas um pouco de contextualização. Ele utiliza comandos simples para que o jogador não tenha dificuldades iniciais para comandar o jogo.

SIGNOS ESTÁTICOS:

- **Canhão:** Representa o personagem principal e controlável do jogo
- **Nave pequena:** Representa perigo ao canhão
- **Nave grande:** Representa o objetivo do jogo, a qual deve ser destruída.
- **Parede amarela:** Representa um objeto de proteção contra os tiros inimigos
- **Estrelas:** Representa o limite superior do jogo
- **Quadrado listrado:** Representa o chão do jogo onde o canhão pode se mover livremente

Metamensagem:

O designer se baseia em elementos que representam alienígenas e objetos espaciais, o que está de acordo com um contexto de invasão alienígena. Estes elementos são bastante coloridos, o que dá a ideia de um jogo divertido e não sério. Além disso, ele oferece uma zona onde o usuário pode se proteger, o que exige que o usuário saiba a hora de atacar e de se defender, ou seja, exige um pouco mais de estratégia básica, pois o canhão deve acertar as naves pequenas antes de poder acertar a nave maior.

SIGNOS DINÂMICOS:

- **Canhão:** Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- **Nave pequena:** O movimento da nave pequena representa um aumento na dificuldade para destruí-la.
- **Nave grande:** O movimento da nave grande representa um aumento na dificuldade para destruí-la.
- **Laser:** O movimento do laser representa ação causal para destruir as naves inimigas
- **Mísseis:** O movimento dos mísseis representa ação causal que pode destruir o canhão
- **Parede amarela:** A mudança da parede amarela representa a diminuição de sua vida

Metamensagem:

O designer utiliza diversos elementos que estão de acordo com a vida real, como mísseis e lasers como objetos de destruição, o que dá uma ideia de que o usuário gosta de jogos em contextos onde o objetivo é proteger o mundo de ameaças alienígenas. Além disso, o jogo apresenta diversos obstáculos que representam perigo ao personagem principal, pois diferentes naves podem soltar lasers ao mesmo tempo, o que torna o jogo mais desafiador e há, também uma área de proteção com vida finita, que permite que o usuário possa se defender e ter um pouco mais de tempo para pensar em sua estratégia.

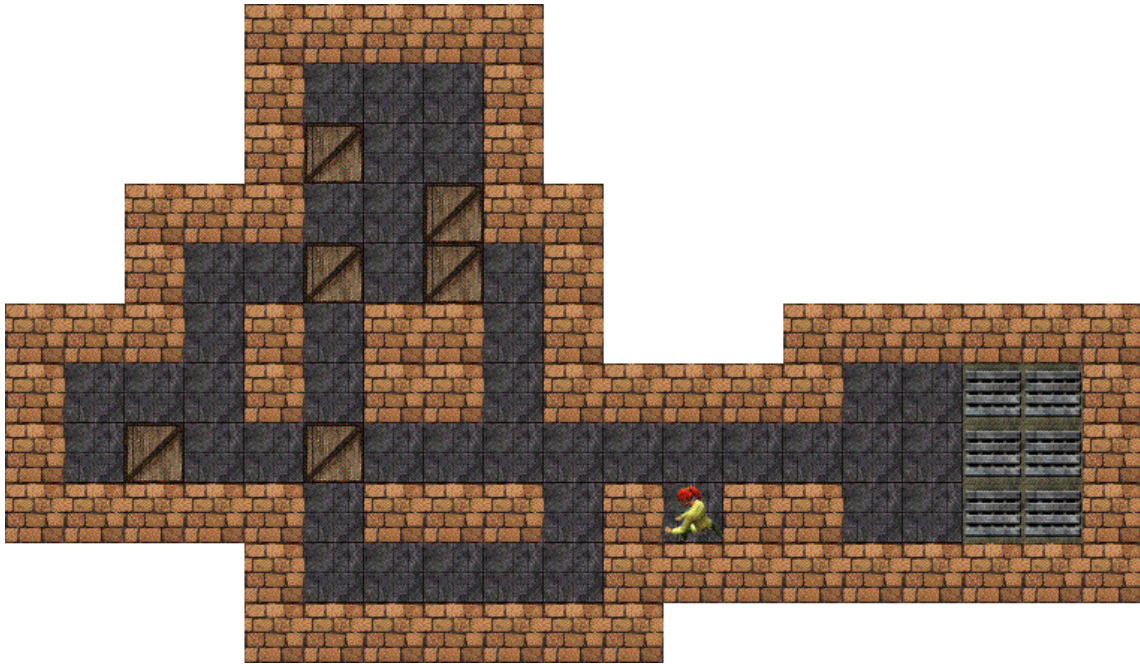
COMPARAÇÃO:

Comparando os signos, pode-se observar a concordância entre todos os três tipos quando se diz respeito ao contexto de invasão alienígena e que o jogo é divertido e cheio de perigos à vida do personagem principal. Não foi observada nenhuma divergência entre os signos deste jogo.

5.1.1.3 Sokoban Original

O terceiro jogo escolhido pela equipe do SGD-Br para ser recriado pelos alunos foi o Sokoban. Neste, o personagem principal é uma mulher que tem como objetivo colocar todas as caixas nos lugares corretos para, assim, concluir o jogo, como pode ser observado na Figura 5.

Figura 5. Imagem do Sokoban Original



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- **Descrição do jogo** (*Você é um repositor que está em um labirinto visto de cima. Você deve empurrar as caixas ao redor e tentar colocá-las em locais designados. Apenas uma caixa pode ser empurrada de cada vez, e as caixas não podem ser puxadas. Quando todas as caixas estiverem no destino, o nível estará completo.*). Faz uma pequena contextualização do jogo, menciona seu objetivo e como conquistá-lo e fala sobre as regras do jogo.
- **Instruções do jogo** (*Use as setas para mover o repositor*): Indica objetivamente os comandos do jogo.

Metamensagem:

O designer assume que o usuário gosta de jogos objetivos que não precisam de uma história detalhada, mas gostam de saber uma pequena contextualização sobre como interagir com o jogo. Além disso, o designer apresenta as regras do jogo na seção de descrição para que o usuário já esteja ciente de como agir em determinadas situações.

SIGNOS ESTÁTICOS:

- **Tijolos:** Representam a parede e o limite de onde o jogador pode andar

- **Ladrilho:** Representa a área em que o personagem pode se locomover livremente.
- **Boneca:** Representa o personagem principal e controlável.
- **Plataforma de metal:** Representa o objetivo do jogo onde as caixas devem ser deixadas.
- **Caixas:** Representam os objetos que devem ser levados ao objetivo do jogo.
- **Parte Branca:** Representa o fundo do jogo

Metamensagem:

O designer assume que o usuário é alguém que prefere jogos estratégicos, pois o cenário não apresenta um tema colorido ou fantasioso e sim um tema e organização mais estratégica, pois o jogo apresenta o formato de um labirinto. Além disso, o jogo pode não apresentar riscos diretos ao personagem, mas riscos que podem impossibilitar que o jogador finalize o jogo.

SIGNOS DINÂMICOS:

- **Boneca:** Com a ação de movimento do jogador, é o personagem principal e controlado do jogo
- **Caixas:** O movimento das caixas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.

Metamensagem:

O designer assume que o usuário gosta de jogos simples, mas com riscos estratégicos, pois o jogo não apresenta nenhum elemento que represente perigo imediato à vida do personagem, apenas ao fato do jogador não conseguir finalizar o jogo. Além disso, o usuário não tem a possibilidade de interagir muito com o cenário, o que dá a ideia de um jogo rápido, mas que precisa de paciência e estratégia, pois as caixas, se colocadas em uma posição errada, não podem mais ser movidas.

COMPARAÇÃO:

Todos os signos apresentam a ideia de um jogo simples e rápido, sem nenhuma divergência. A mensagem de que é um jogo de raciocínio lógico é passada prioritariamente pelos signos dinâmicos e estáticos, mas também pelos metalinguísticos, mostrando uma consistência entre eles. Além disso, podemos observar que os signos metalinguísticos trazem

as regras do jogo, facilitando jogadores iniciantes de não se perderem durante a execução do jogo, reforçando a ideia de um jogo rápido que não necessita que o usuário perca tempo aprendendo sobre o jogo durante sua execução.

5.1.2 Execução e Análise do MIS dos jogos dos alunos

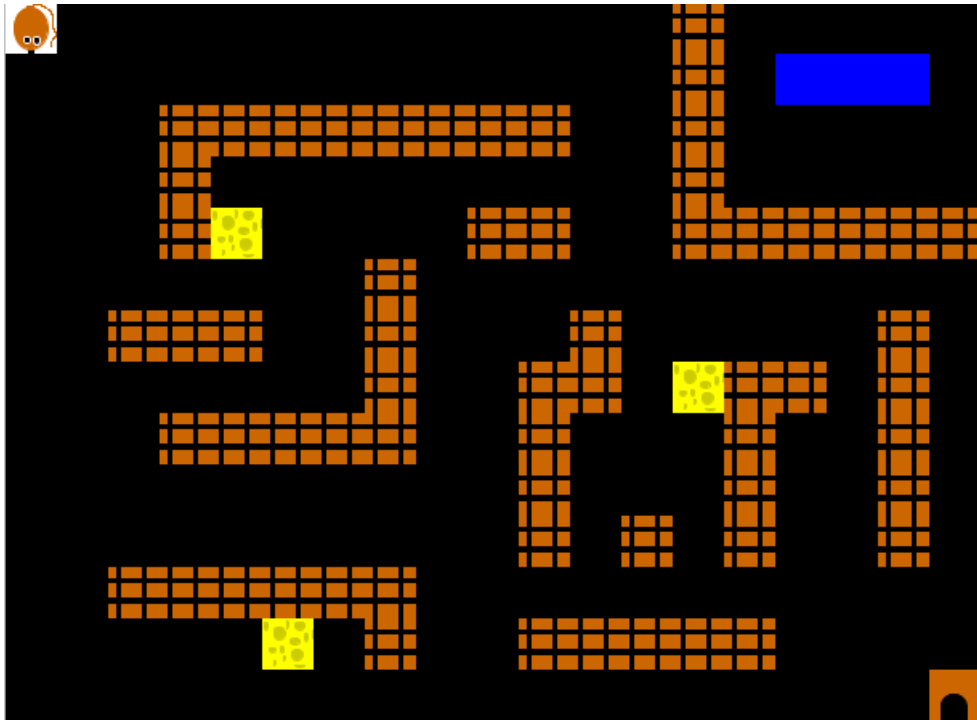
Como citado anteriormente, os jogos desenvolvidos pelos alunos se encontram na plataforma do PoliFacets, onde podemos visualizá-los e executá-los sem que seja necessário o uso do AgentSheets.

No total foram analisados 27 jogos, o que tornou o MIS muito extenso e, por esta razão, o resultado de todos os MIS realizados encontram-se no Apêndice A desta pesquisa. Outro fato observado foi que muitos dos alunos reutilizaram as imagens dos jogos originais em seus jogos, mantendo os mesmos signos estáticos e/ou dinâmicos, alterando apenas os signos metalinguísticos. Ainda assim, se fez importante analisar os signos de todos os jogos separadamente, pois, muitas vezes, a disposição dos signos no mapa de *tiles* (“mundo” do jogo) muda o significado da metamensagem, então referenciamos os signos que se repetiram e criamos novas metamensagens para aqueles jogos cuja organização destes signos repetidos, no mapa de *tiles*, apresentou um significado diferente para o jogo.

Após a realização dos MIS, além de termos obtido lucidez sobre os jogos desenvolvidos pelos alunos do projeto e a mensagem que estes tentaram transmitir através de seus jogos, pudemos observar diversas diferenças e semelhanças entre os jogos originais e os jogos desenvolvidos pelos alunos:

- Aumento do número de níveis do jogo, o que aconteceu na grande maioria dos jogos desenvolvidos pelos alunos e que não existia nos jogos originais.
- Criação de novos signos e elementos gráficos, como pode ser visto na figura 6.

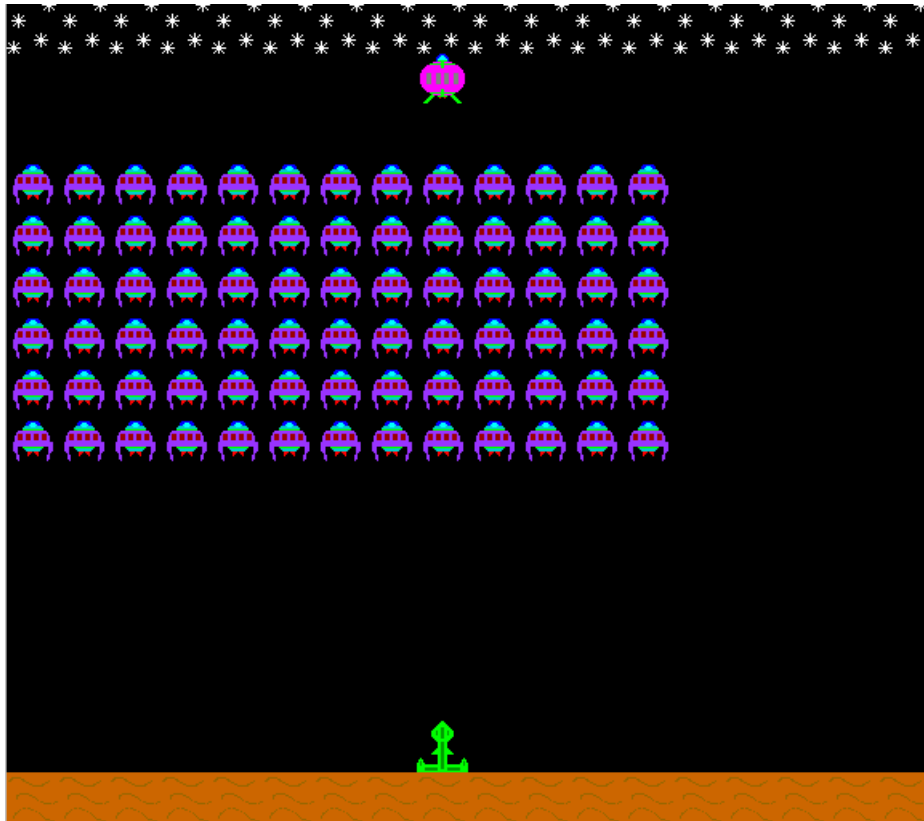
Figura 6. Sokoban Desenvolvido pelo aluno A06



Fonte: PoliFacets

- Nenhum dos jogos desenvolvidos tem um objetivo final diferente dos jogos originais.
- Adição de contadores no Sokoban, como no A2-Sokoban e no A12-Sokoban do Apêndice A.
- Remoção das proteções no Space Invaders desenvolvidos pelos alunos como mostrado na figura 7.

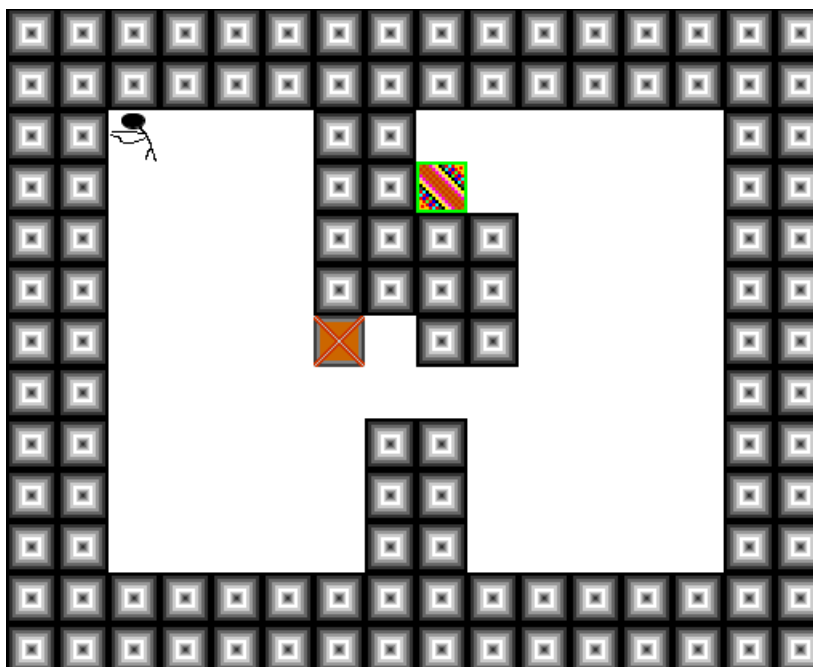
Figura 7. Space Invaders Desenvolvido pelo aluno A02



Fonte: PoliFacets

- Diferença de contextos em jogos dos três tipos, como mostram os Sokobans das figuras 6 e 8, onde o primeiro tem um rato querendo levar o queijo para casa e no outro um homem pondo as caixas em seus respectivos lugares.

Figura 8. Sokoban Desenvolvido pelo Aluno A07



Fonte: PoliFacets

- Alguns alunos modificaram os controles dos jogos, mudando das setas para as letras A, W, S, e D, que é um padrão utilizado por outros jogos.
- Adição, remoção e edição de mensagens nos jogos, o que ocorreu na grande maioria dos jogos.
- Aumento ou diminuição do nível de dificuldade, principalmente quando há mais de um nível no jogo.

5.2 Realização de Testes Funcionais nos Jogos dos Alunos

Para realização dos testes funcionais, era necessário que nós tivéssemos algum tipo de documentação para tomarmos como base e saber como o sistema ou os jogos deveriam reagir dada uma entrada específica. No entanto, como essa documentação não foi criada, já que a atividade de testes não fazia parte do roteiro do projeto SGD-Br, utilizamos os resultados do MIS dos jogos originais, onde entendemos as metagensagens e o que os desenvolvedores queriam passar com elas como fonte de conhecimento das funcionalidades do sistema. Assim, os jogos originais funcionaram como uma espécie de “especificação de requisitos” para os jogos criados pelos alunos. Na Tabela 2, podemos ver, de forma geral, o roteiro dos testes realizados nesta etapa.

Tabela 2. Roteiro Geral de Testes dos Jogos

Jogo	Testes Realizados
Frogger	<ul style="list-style-type: none"> • Passar pelos obstáculos laterais • Andar por cima e ao redor dos caminhões • Andar pela água e pelos objetos da água • Andar nos <i>tiles</i> em que o caminhão é gerado ou excluído no mapa de <i>tiles</i>
Space Invaders	<ul style="list-style-type: none"> • Destruir naves inimigas em todas as alturas • Destruir a própria defesa • Destruir a nave mãe • Destruir o tiro inimigo com o laser
Sokoban	<ul style="list-style-type: none"> • Mover caixas para os cantos das paredes • Empurrar uma caixa em direção à outra
Comum aos 3 jogos	<ul style="list-style-type: none"> • Análise do tamanho do mapa de <i>tiles</i> • Locomoção do Personagem Principal • Andar pelas extremidades do jogo • Finalizar o jogo • Observar Bugs Visuais • Concluir e mudar de nível

Fonte: Elaborada pelo autor

Com o conhecimento sobre as funcionalidades do sistema, realizamos testes onde dávamos entradas específicas, como um movimento, uma ação ou simplesmente aguardávamos para ver quais eventos temporais poderiam vir a acontecer. Vários destes bugs podem ser vistos na tabela 3, que apresenta os bugs encontrados com maior frequência durante a busca. Todos os bugs encontrados em seus respectivos jogos podem ser encontrados no Apêndice B.

Tabela 3. Bugs constantes encontrados durante os testes funcionais

Jogo	Bugs
Frogger	<ul style="list-style-type: none"> • Problemas com a velocidade dos caminhões.

	<ul style="list-style-type: none"> • Sapo não morre ao ir para cima do caminho, apenas à frente dele. • Andar com o sapo em cima do <i>tile</i> gerador dos caminhos impede que o jogo gere novos caminhos. • Caminhões não são excluídos do cenário ao chegar no túnel • Sapo atravessando obstáculos laterais • Troncos e Tartarugas não sendo gerados
Space Invaders	<ul style="list-style-type: none"> • Canhão não consegue destruir naves que estão no <i>tile</i> acima do dele • Naves saindo dos limites da tela
Sokoban	<ul style="list-style-type: none"> • Problemas com as extremidades do jogo, permitindo que o usuário saia do cenário. • Caixas posicionadas de forma que impossibilitam sua movimentação
Comum aos 3 jogos	<ul style="list-style-type: none"> • Jogos continuam sendo executados mesmo depois de finalizados • Tamanho incorreto dos mapas de <i>tiles</i> • Jogos iniciando a partir do nível 2 ou 3 • Jogos finalizados no nível 1, mesmo existindo outros níveis • Jogo sem agente controlável • Níveis iniciando já finalizados • Bugs visuais • Bugs de animação

Fonte: Elaborado pelo autor

5.3 Análise dos Testes dos Alunos

A análise dos quarenta e seis resultados dos testes dos alunos foi feita a partir da transcrição do que havia sido escrito pelos próprios alunos, num documento contendo as

perguntas e suas respostas. Logo após, fizemos uma tradução e interpretação dos resultados dos testes feitos pelos alunos e escrevemos na parte que chamamos de Análise do aluno, que se encontra em cada análise de teste no Apêndice B.

Com os resultados dos testes dos alunos transcritos, traduzidos e organizados, escrevemos os problemas encontrados pelos testes funcionais na seção de Bugs encontrados, onde foram escritos todos os bugs encontrados durante a realização dos testes funcionais, pois assim seríamos capazes de realizar uma comparação entre os resultados dos alunos e os resultados encontrados durante a fase de testes funcionais.

Após a análise dos testes dos alunos, pudemos observar diversos tipos de bugs apontados pelos alunos, como mostrado a seguir:

- Falta de signos nos jogos, seja o personagem principal, um objeto importante ou o objetivo final do jogo.
- Problemas de mudança de um nível para o outro.
- Jogos iniciando ou reiniciando no segundo nível.
- Problemas com o tamanho dos mapas de *tiles* dos jogos, impossibilitando que o aluno visse o cenário completo.
- Problemas de movimentação do signo controlável, ou personagem principal do jogo.

Assim como problemas que geralmente não eram relatados pelos alunos:

- Ao finalizar, o jogo continua sendo executado.
 - Este não era um bug que impedia o usuário de jogar, mas o jogo continuava sendo executado mesmo depois de uma vitória ou derrota.
- Bugs temporais, que acabavam aparecendo mesmo sem o usuário interagir com o jogo, como o problema do “engarrafamento” que apareceu no Frogger, onde os caminhões não eram excluídos do mapa de *tiles* ao chegar ao final da pista.
- Houve um bug que foi encontrado em todos os Froggers, onde o sapo, ao ficar no *tile* à frente do túnel de saída dos caminhões, impedia que mais caminhões fossem criados, mas nenhum aluno reportou esse problema.
- Outro bug que não foi relado por nenhum dos alunos aparece nos Space Invaders e ocorre ao tentar destruir uma nave inimiga que está no *tile* acima do canhão, pois os tiros são criados no segundo *tile* acima deste, impossibilitando a destruição destas naves que estão no *tile* acima do canhão.

- Bugs onde o jogador pode atravessar a parede.
- Bugs visuais na interface.
- Frogger passando por cima dos obstáculos.
- Alguns níveis dos jogos já iniciam finalizados, passando direto para o segundo nível.
 - Neste caso, a maioria dos alunos não percebeu esse detalhe e reportou como se o jogo estivesse iniciando no segundo nível.

Alguns testadores deram sugestões de melhoria para os jogos dos colegas, em geral, as melhorias falavam sobre os seguintes pontos:

- Modificação da dificuldade do jogo para mais fácil ou mais difícil.
- Melhorar o visual do jogo.
- Aumentar o número de níveis do jogo.

Também houve problemas que foram confundidos e relatados como bugs:

- Alguns Sokobans tiveram um nível de dificuldade maior que os outros e alguns alunos reportaram isso como um bug.
- A falta de mensagens informando a finalização do jogo também foi relatada como um bug, quando deveria aparecer como uma sugestão.
- Sugestões foram escritas na área onde deveriam ser descritos os bugs.

5.4 Consolidação dos Resultados

A análise dos testes nos traz uma diversidade de problema ou bugs que foram relatados ou não pelos alunos. Ao julgar estes resultados, observamos que a grande maioria dos testes que eram relatados pelos alunos fazia parte ou atrapalhava “o caminho feliz” do software, ou seja, as ações previstas. Isso acaba por mostrar que grande parte dos alunos não se deu ao trabalho de testar outros caminhos, por exemplo, passar por cima dos obstáculos com o sapo no jogo Frogger, ou tentar atravessar a parede no Sokoban.

Para entender melhor o comportamento e as motivações dos alunos, faz-se necessária uma análise mais profunda para justificar a qualidade dos resultados e identificar hipóteses para possíveis problemas encontrados durante a análise dos testes realizados pelos alunos. Neste ponto, a coleta de dados se dá principalmente pelo formulário onde os alunos escreveram as respostas dos testes dos jogos e entrevista com um participante do projeto.

Com esse objetivo em vista, criamos três categorias gerais de testes, os quais foram baseados nos seguintes critérios:

1. Informar que havia bugs, ou não, no jogo. Neste primeiro ponto, vale ressaltar que dos 46 testes, 42 deveriam apresentar bugs, ou seja, 92% dos testes realizados pelos alunos deveriam informar a presença de bugs nos jogos.
2. Detalhamento do bug, em qual fase do jogo encontrá-lo, em qual elemento do jogo está o problema, qual razão de ocorrer o bug.
3. Correção do bug, o que o desenvolvedor pode fazer para corrigir o bug encontrado pelo testador.
4. Sugestão que pode melhorar de alguma forma o jogo, seja ela sobre dificuldade, gráfico ou jogabilidade.

Testes efetivos

Nesta categoria, foi levado em consideração o quanto o aluno se aprofundou ao relatar o bug, informando detalhes como onde o bug estava ou qual o agente que apresenta o bug, o que é o bug e o caminho que leva ao bug, além disso, o aluno também informa como corrigi-lo. Esta categoria leva em consideração que, caso o aluno tenha feito alguma sugestão, esta deve se encaixar no quarto critério. Testes de jogos os quais não encontramos bugs e o aluno também não relatou nenhum bug também se encontram nesta categoria. Apenas 8 dos 46 testes realizados se enquadram nesta categoria, com um total aproximado de 17%.

Testes efetivos, mas com poucas informações

Nesta categoria, encontram-se testes que, apesar de apresentar os bugs, fornecem poucas ou nenhuma informação sobre o bug, como definição, onde encontrá-lo ou qual caminho necessário para que o bug ocorra. Também não é fornecida nenhuma solução para a correção do bug, mas podem possuir alguma sugestão relevante para o desenvolvedor. Dos 46 testes analisados, 18 se encaixam nesta categoria, sendo um total de 39%.

Testes inefetivos

Alguns dos testes analisados, apesar de escritos, não mostravam resultados satisfatórios. Nestes testes, mesmo com o jogo apresentando algum tipo de bug, o testador informa que não há bugs ou que há bugs mesmo sem ter e podem, ou não, ter alguma sugestão

relevante para o desenvolvedor. Dos 46 testes analisados, 20 se encaixam nesta categoria, totalizando cerca de 43%.

Também foram encontradas situações durante a análise que se encaixam em todos os tipos de testes:

- Alguns alunos utilizaram a área de sugestões para reforçar o que estava escrito na parte de apresentação de bugs
- Alguns testes apresentam elogios ou críticas não construtivas na seção de sugestões, como, por exemplo, “*yea, fix it*” (Sim, conserte-o!).
- Alguns alunos escreveram a correção ou detalhamento do bug na área de sugestões
- Houve relatos de bugs que na verdade eram apenas obstáculos para aumentar a dificuldade do jogo e que não implicavam na jogabilidade, como, por exemplo, uma caixa do Sokoban em um local que limitava sua movimentação.
- Enquanto alguns alunos preferiram usar uma linguagem formal para fazer seus relatos, outros utilizaram uma linguagem informal e gírias, o que pode facilitar a comunicação entre os alunos, visto que os resultados dos testes eram voltados para os desenvolvedores dos jogos, que são os próprios alunos. No entanto, esse tipo de linguagem mais intimista também pode vir a dificultar o entendimento da informação que o aluno estaria passando para pessoas de fora.

Como pôde ser observado, mais da metade dos testes realizados pelos alunos entraram nas categorias de testes efetivos, mostrando-se assim como um ótimo resultado, levando em conta que os alunos não foram preparados e nem tiveram nenhuma educação formal para a realização de atividades de testes, seguindo sua intuição e contando com suas experiências desenvolvendo seus próprios jogos, focando na qualidade dos jogos analisados e não na qualidade dos testes em si. No entanto, quase metade dos testes se encaixa na categoria de testes inefetivos, pois estes não passam informações concretas ou não passam nenhuma informação sobre os bugs dos jogos. Quanto a isso, podemos levar em consideração determinados pontos que podem ter interferido na qualidade dos resultados dos testes.

- O primeiro e mais importante ponto é que os alunos são usuários finais, eles não foram treinados e, neste caso, não estudaram para fazer esse tipo de atividade. Além disso, como foi uma atividade solicitada pelo professor, que também é usuário final, a equipe do projeto não realizou nenhum tipo de treinamento ou deu alguma informação sobre a atividade de testes.

- São crianças, com média de treze anos de idade.
- A atividade era opcional e não seria levada em conta no resultado escolar do aluno.

De acordo com Ko et al (2011), uma das abordagens comumente utilizadas por usuário finais é o que chamamos de *Oracles and Overconfidence* (Oráculos e excesso de confiança). De acordo com essa abordagem, a questão central para qualquer tipo de verificação é a decisão sobre se um determinado comportamento ou saída do programa está correto. A fonte de tal conhecimento é geralmente referida como um oráculo, que pode ser uma pessoa, tomando decisões mais ou menos formais sobre a correção do comportamento do programa (KO et al, 2011). Pessoas não são perfeitas e, dito isto, resultados baseados no que um testador julga por correto, principalmente sendo este um usuário final, sem nenhum tipo de documentação para tomar como base e que não tem amplo conhecimento e experiência na área, podem gerar resultados insatisfatórios, principalmente quando o testador está analisando um programa desenvolvido por outra pessoa, como é a realidade dos alunos do caso analisado nesta pesquisa, que desenvolveram seus jogos e testaram os jogos dos colegas baseados no que eles mesmos haviam desenvolvido.

6 CONSIDERAÇÕES FINAIS

Nesta pesquisa, realizamos uma análise das atividades de testes no contexto do projeto Scalable Game Design Brasil, utilizando-se um dos métodos da Engenharia Semiótica, o MIS, além da realização de testes funcionais para facilitar a inspeção dos jogos criados por alunos, que são identificados como usuários finais. É importante ressaltar a relevância deste tipo de análise, pois o crescimento de sistemas desenvolvidos por usuários finais é uma realidade, e técnicas e abordagens que facilitem seus trabalhos podem melhorar a qualidade do produto final.

No encerramento desta pesquisa, foi visto como os usuários finais enxergam o mesmo problema de maneiras diferentes, pois em um bug em determinado jogo, diferentes alunos deram diferentes soluções, mas dizer a que está correta ou que se aproxima melhor da realidade do produto cabe apenas a pessoa que irá utilizá-lo, pois ela sabe quais as funções do sistema serão úteis para o objetivo de sua criação.

Portanto, espera-se que, com os resultados obtidos nas análises apresentadas, futuros projetos baseados nesse tipo de contexto abranjam uma área maior da EUSE, pois a necessidade da atividade de testes, neste caso, foi sentida pelo professor da turma analisada, que também se encontra na posição de usuário final, e alguns problemas encontrados poderiam ter sido evitados caso houvesse uma capacitação do professor, visto que este era a pessoa com a qual os alunos tiravam suas dúvidas.

Além disso, também vemos como contribuição de nossa pesquisa, diversos exemplos de aplicações do MIS em jogos, assim como os resultados dos testes funcionais realizados neles.

7 TRABALHOS FUTUROS

Nesta seção apresentamos certos pontos que podem ser melhorados em pesquisas futuras com contextos parecidos ao caso abordado, além de apresentar novas oportunidades de pesquisas que poderiam derivar desta e de pesquisas referenciadas neste trabalho.

O primeiro ponto a ser ressaltado é o fato de não terem sido apresentados outras fases da EUSE para os alunos, apenas a parte de desenvolvimento, o que acaba por criar um pensamento onde o usuário final foca apenas no término do produto, esquecendo a importância da qualidade deste. Desta forma, vemos que, para trabalhos futuros, é importante ressaltar, mesmo que de forma superficial, a importância da EUSE e como a qualidade do produto final é impactada com sua metodologia. Além disso, o professor, que também é um usuário final, poderia ter sido melhor capacitado para lidar com suas atividades no âmbito da EUSE, o que poderia ter melhorado a estrutura dos testes realizados pelos alunos.

Nesta pesquisa, focamos na atividade de testes realizadas pelos alunos, mas esta não foi a única atividade realizada por eles. Tendo em vista que estes desenvolveram seus próprios jogos, como trabalho futuro, sugerimos uma análise de código em busca de padrões e técnicas de desenvolvimento utilizadas pelos alunos. Além disso, seria muito interessante ter visto uma atividade onde os alunos liam os resultados dos testes dos colegas e tentavam corrigir os problemas citados.

REFERÊNCIAS

BARBOSA, S. D. J; SILVA B. S. **Interação Humano-Computador**. [S.l]: Elsevier, 2010.

BRANDÃO, A; TREVISAN, G. D; BRANDÃO, L; MOREIRA, B; NASCIMENTO, G; MOURAÃO, P. T; VASCONCELOS, C. N; CLUA, E. G; CLUA, L. R. Semiotic Inspection of a Game for Children with Down Syndrome. **Games and Digital Entertainment** (SBGAMES), 2010.

BURNETT, M. M; SCAFFIDI, C. **The Encyclopedia of Human-Computer Interaction**, 2nd Ed. 2014. Aarhus, Denmark: The Interaction Design Foundation. Disponível em: https://www.interaction-design.org/encyclopedia/enduser_development.html. Acesso em 08 jun. 2015.

COGNIFIT, **O que são habilidade cognitivas?** 2015. Help Center. Disponível em: http://support.cognifit.com/customer/pt_br/portal/articles/594172-o-que-s%C3%A3o-as-habilidades-cognitivas-. Acesso em 08 jun. 2015.

COUTINHO, P. H. M. **Módulo de Teste de Software**. Escola Superior Aberta do Brasil, 2011.

DE SOUZA, C. S; LEITÃO C. F; PRATES R. O; DA SILVA E. J; The semiotic inspection method, **Proceedings of VII Brazilian symposium on Human factors in computing systems**, November 19-22, 2006, Natal, RN, Brazil.

KO, A. J; ABRAHAM, R; BECKWITH, L; BLACKWELL, A; BURNETT, M; ERWIG, M; SCAFFIDI, C; LAWRENCE, J; CSAIL, M; LIEBERMAN, H; MYERSB; ROSSON, M. B; ROTHERMEL, G; SHAW, M; WIEDENBECK, S. The State of the Art in End-User Software Engineering. **Journal ACM Computing Surveys** (CSUR) Vol. 43, n. 3, Abr., 2011.

MONTEIRO, I. T; SALGADO, L. C. C; MOTA, M. P; SAMPAIO, A. L; DE SOUZA, C. S. Signifying software engineering to computational thinking learners with AgentSheets and PoliFacets. **Journal of Visual Languages & Computing**, Vol. 40, p. 91-112, Jun., 2017.

POLIFACETS, **Explore várias facetas dos jogos e simulações feitos no AgentSheets**. 2012. Portal do projeto. Disponível em: http://www.serg.inf.puc-rio.br/PoliFacets/index.lua?lg=pt_br. Acesso em 14 mai. 2015.

SALGADO, L. C. C. **CommEST – Uma ferramenta de apoio ao método de Avaliação de Comunicabilidade**. Dissertação de Mestrado. PUC (Pontifícia Universidade Católica). Rio de Janeiro, 2007.

SCALABLE GAME DESIGN, **SCALABLE GAME DESIGN WIKI**, 2009, Portal do Projeto. Disponível em: <http://sgd.cs.colorado.edu/wiki/Scal>. Acesso em 13 jan. 2015.

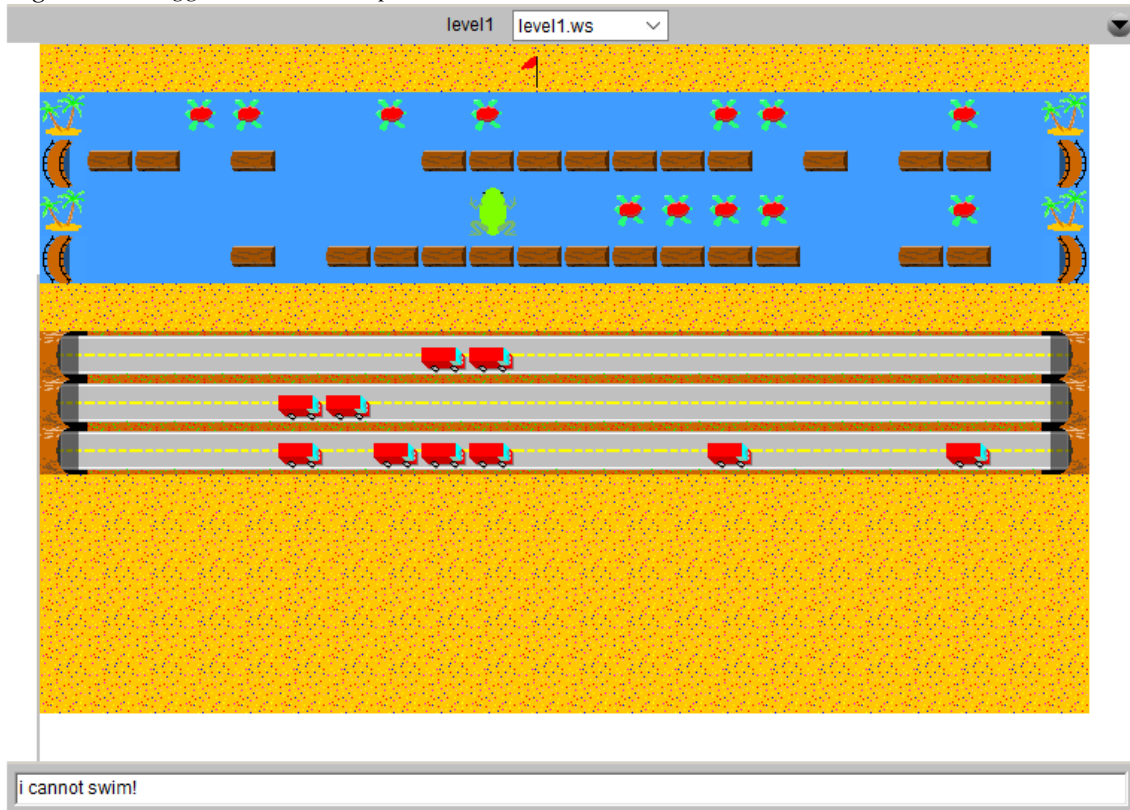
SCALABLE GAME DESIGN BRASIL, **BEM VINDOS AO SCALABLE GAME DESIGN BRASIL**, 2011, Portal do projeto. Disponível em: http://www.serg.inf.puc-rio.br/wiki/index.php/P%C3%A1gina_principal. Acesso em: 21 abr.2015.

APÊNDICE A – Resultado dos MIS dos jogos dos alunos

Nesta seção, são apresentados os resultados dos MIS realizados em todos os jogos desenvolvidos pelos alunos.

A01 – FROGGER

Figura 9. Frogger Desenvolvido por A01



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Mensagem ao tentar passar por cima da ponte (*i cannot walk over – Não consigo andar aqui por cima*): Avisa que o usuário não pode utilizar a ponte para atravessar o rio.
- Mensagem ao cair na água (*I Cannot swim – Eu não sei nadar*): Avisa a morte do sapo ao cair na água.
- Descrição do jogo (*this game is fun for kids of all ages try to get to the flag without getting hit by the trucks and not touching the water!* – Este jogo é divertido para crianças de todas as idades, tente chegar à bandeira sem ser atingido pelos caminhões e sem tocar na água): Deixou explícito os perigos representados pela água e pelos caminhões e o objetivo do jogo que é chegar à bandeira.
- Instruções do jogo (*Use the up, down, left side and right side arrows to go through the road and river. if you immediately make contact with the trucks or water you will die. As soon as you get onto the flag you will be sent to the next thrilling level!* – Use as setas direcionais para cima, para baixo, para a direita e para a esquerda para mover-se pela estrada e pelo rio. Se você fizer contato imediato com os caminhões ou a água, você irá morrer. Quando você chegar na bandeira, você será enviado para um próximo nível sensacional!): Além de especificar os controles do jogo, reforça os perigos representados pela água e pelos caminhões e o objetivo do jogo, que é chegar à bandeira e ir para o próximo nível.

- Mensagem ao tentar subir na ponte/ilha/túnel (*I cannot walk over*): Mensagem que informa que o jogador não pode passar por cima destes objetos.
- Mensagem ao cair na água (*I cannot swim*): Mensagem de morte ao cair na água.
- Mensagem ao chegar na bandeira do segundo nível (*I made it*): Mensagem de vitória ao chegar ao objetivo.

Metamensagem:

Na visão do designer, o usuário é uma criança que gosta de se divertir, com um objetivo pré-determinado e que consegue ler em outro idioma (inglês) para entender o jogo. Como é comum em muitos jogos, neste também há ações proibidas e situações de perigos presentes. O jogo tem seu uso facilitado, através de comandos simples como as setas do teclado. Ele também se baseia em um usuário que gosta de receber feedbacks quando ocorre algo inesperado no jogo como morte do personagem principal ou andar por locais proibidos e se esforça para passar a um nível mais desafiador.

SIGNOS ESTÁTICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idênticos aos do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

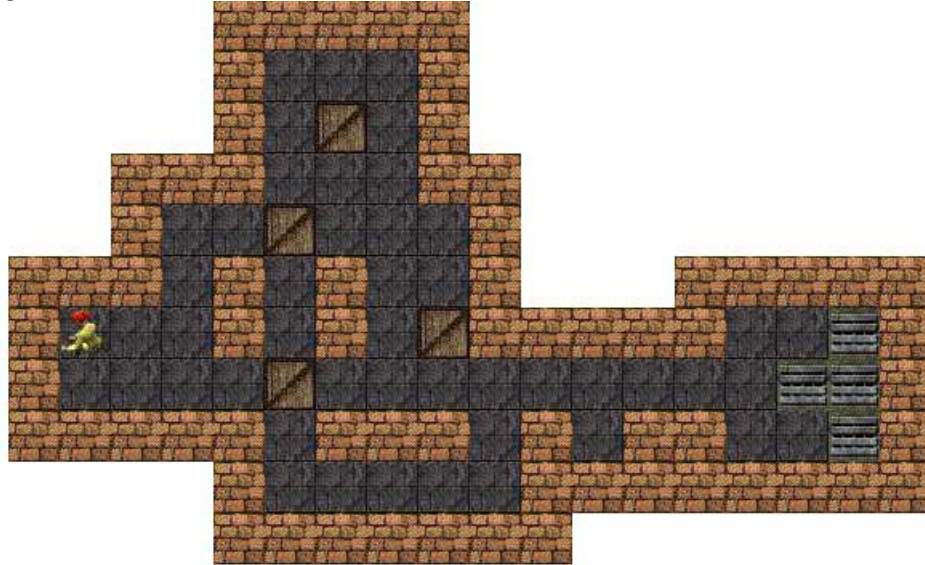
Na visão do designer, o usuário é alguém que gosta de ter tempo de pensar e pode voltar os obstáculos pois o sapo pode andar para trás e voltar nos obstáculos. O jogo apresenta um nível mais difícil onde os caminhões, tartarugas e troncos são mais rápidos, aumentando as chances de cair na água ou ser atropelado, o que leva a pensar que o usuário gosta de níveis mais desafiadores.

COMPARAÇÃO:

Nos três tipos de signos os elementos do jogo são apresentados com o mesmo sentido, sem nenhuma divergência. Além disso, quando o designer deixa claro que o usuário é uma criança, ele faz uso de signos estáticos e dinâmicos que facilitam o entendimento do jogo, já que ele usa elementos vistos no dia-a-dia da criança e não troca o sentido deles.

A01 - SOKOBAN

Figura 10. Sokoban Desenvolvido por A01



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*the meaning of the game is to drag the boxes one by one onto the metal platform.*): Menciona o objetivo do jogo.
- Instruções do jogo (*use the up, down, right and left arrows to move the box around until it its in the metal platform*): Menciona o objetivo do jogo e os controles.

Metamensagem:

O designer assume que o usuário gosta de jogos objetivos que não precisam de uma história detalhada. O importante mesmo é conhecer os comandos para se mover e ganhar o jogo.

SIGNOS ESTÁTICOS:

Idênticos aos do Sokoban Original da seção 5.1.1.3

Metamensagem:

Idênticos aos do Sokoban Original da seção 5.1.1.3

SIGNOS DINÂMICOS:

Idênticos aos do Sokoban Original da seção 5.1.1.3

Metamensagem:

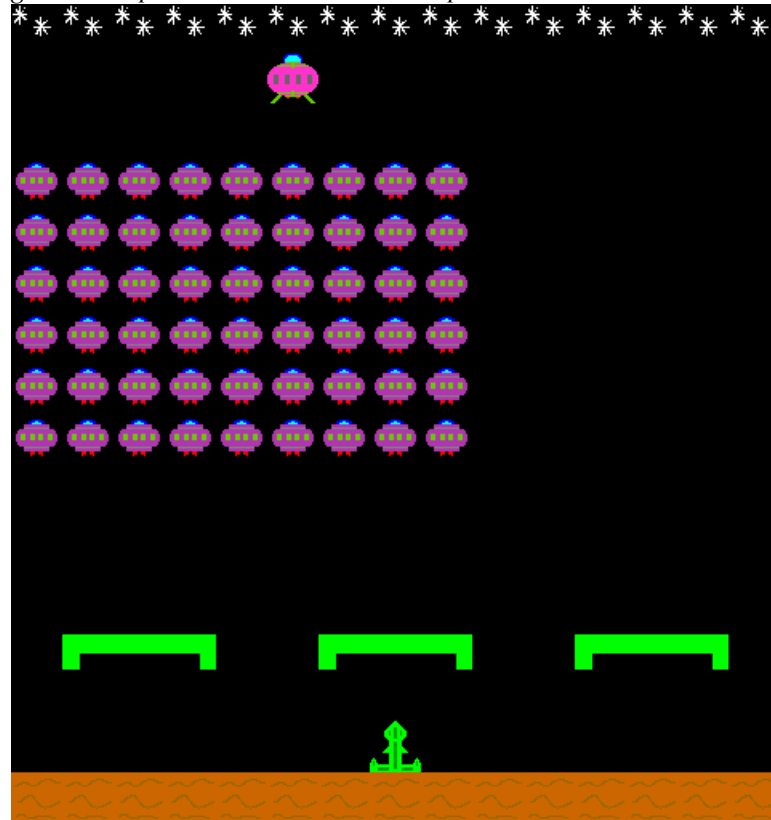
Idênticos aos do Sokoban Original da seção 5.1.1.3

COMPARAÇÃO:

Todos os signos apresentam a ideia de um jogo simples e rápido, sem nenhuma divergência. A mensagem de que é um jogo de raciocínio lógico é passada prioritariamente pelos signos dinâmicos, mas também pelos metalinguísticos, mostrando uma consistência entre eles. Esta mensagem não é passada pelos signos estáticos. Portanto, o usuário precisa interagir com o jogo para conhecer seus desafios.

A01 - SPACE INVADERS

Figura 11. Space Invaders Desenvolvido por A01



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*try to defeat the space aliens by shooting down the mother ship and the aliens!*): Mostra um pouco de narrativa e o objetivo do jogo.
- Instruções do jogo (*use the right and left keys to move side to side and use the spacebar to shoot the aliens down! try not to hit the your wall!*): Menciona os controles do jogo e dá uma dica sobre o jogo.
- Mensagem de erro ao atingir a parede de proteção: Mostra um bug do jogo onde a plataforma não sabe como reagir à ação.

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com histórias simples, permitindo que ele inicie o jogo rapidamente, utilizando comandos básicos para movimentar e atirar.

SIGNOS ESTÁTICOS:

Idênticos aos do Space Invaders Original da seção 5.1.1.2

Metamensagem:

Idênticos aos do Space Invaders Original da seção 5.1.1.2

SIGNOS DINÂMICOS:

Idênticos aos do Space Invaders Original da seção 5.1.1.2

Metamensagem:

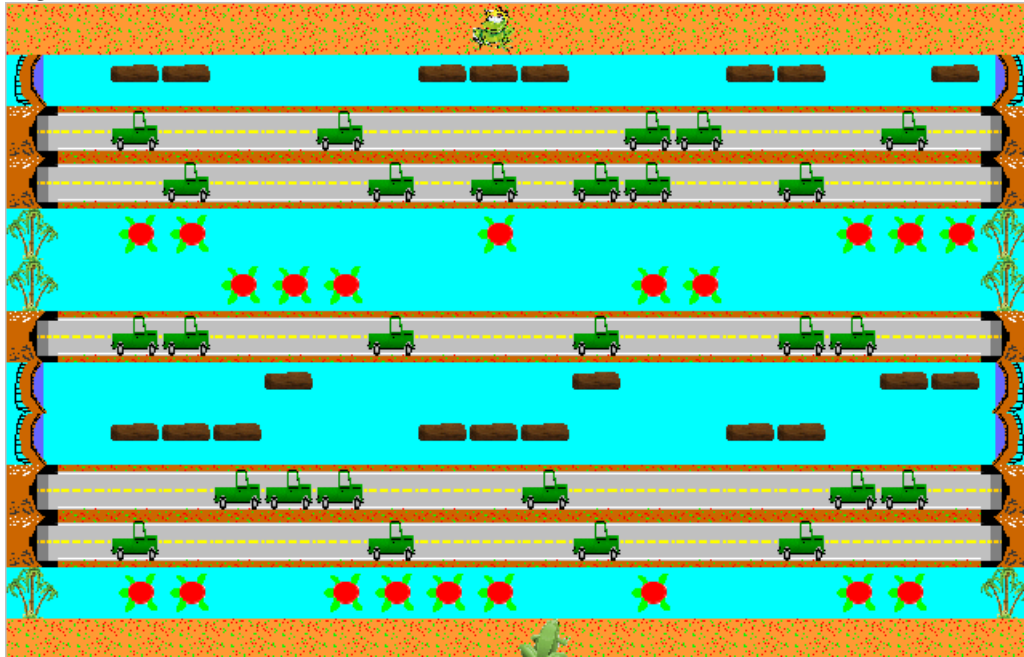
Idênticos aos do Space Invaders Original da seção 5.1.1.2

COMPARAÇÃO:

Nos signos metalinguísticos, o designer passa a ideia de um jogo simples e rápido, porém, nos signos dinâmicos e estáticos, percebe-se que o jogo exige um pouco mais de estratégia para se chegar ao objetivo.

A02 - FROGGER

Figura 12. Frogger Desenvolvido por A02



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Poor Froggy has been separated from his girlfriend, Frogette. To get to his girlfriend, he must get to the flag first, then to Frogette.*): Representa a narrativa e o objetivo do jogo, que é encontrar a Frogette.
- Instruções do jogo (*Use the arrow keys to control Froggy. Get to the flag on the other side of the roads and river and then get to Frogette. Be careful! Trucks can kill you, and this frog can't swim.*): Menciona os controles necessários para jogar, o objetivo do jogo e deixa claro os perigos do jogo, que são os caminhões e a água.
- Aviso ao cair na água (*I cannot swim*): Avisa a morte do jogador por cair na água.
- Aviso ao chegar na Frogette (*Oh, darling! I missed you! Thanks for saving me.*): Aviso de vitória ao chegar ao objetivo (Frogette).

Metamensagem:

O designer se baseia em um usuário que gosta de uma história mais elaborada, mesmo que para um jogo simples, o objetivo é salvar a namorada e não somente passar de fases. O designer também vê o usuário como alguém que gosta de desafios, colocando diversos obstáculos para atrapalhar a conclusão do objetivo. O jogo oferece comandos simples e mensagens de feedback ao cumprir o objetivo ou morrer.

SIGNOS ESTÁTICOS:

- Sapo: Representa o personagem principal do jogo, o que é controlado pelo jogador.
- Estrada: Representa o caminho dos caminhões e onde o sapo deve passar para chegar ao objetivo.
- Água: Representa um local de perigo e morte para o sapo.
- Ponte: Representa o local de onde os troncos aparecem.
- Túnel: Representa o local de onde os caminhões saem.
- Tronco: Representa o objeto que ajuda o sapo a atravessar a água.
- Grama: Representa um local seguro onde o sapo pode se locomover.
- Caminhão: Representa um objeto de perigo e morte ao sapo.
- Sapa: Representa o principal objetivo do jogo, onde o sapo deve chegar.
- Bandeira: Representa o objetivo inicial do jogo, onde o sapo deve chegar.
- Tartaruga: Representa o objeto que ajuda o sapo a atravessar a água.
- Ilha: Representa o local de saída e chegada das tartarugas.

Metamensagem:

O jogo utiliza elementos coloridos o que dá uma ideia de que o usuário é uma criança. Além disso, todos os elementos utilizados são baseados em objetos reais, vistos no dia-a-dia, o que simplifica o entendimento do usuário sobre o que se deve fazer no jogo.

SIGNOS DINÂMICOS:

- Sapo: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo
- Tartaruga: O movimento da tartaruga representa ação causal que pode matar o sapo
- Caminhão: O movimento do caminhão representa ação causal que pode matar o sapo
- Tronco: O movimento do tronco representa ação causal que pode matar o sapo

Metamensagem:

O jogo representa elementos simples e de fácil entendimento, fazendo uso de objetos que podem realmente apresentar perigos para um sapo na vida real e o que ele pode utilizar para ajudá-lo, que também ajuda a observar que o usuário é alguém que gosta de dificuldade para chegar ao objetivo.

COMPARAÇÃO:

Ao comparar os signos, podemos observar que todos dão a ideia de um jogo simples e de fácil entendimento. O designer utiliza diversas mensagens de feedback e uma narrativa de vitória, onde a própria “sapa” dá os parabéns ao jogador, reforçando que ele realmente conseguiu atingir o objetivo do jogo, que era salvá-la. Não foi vista nenhuma divergência entre os signos.

A02 - SOKOBAN

Figura 13. Space Invaders Desenvolvido por A02



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Wall-E is having trouble cleaning up again. He has to get the crate to the red box, so it can be disposed of. Once he arrives, the box sends him to another world - with the crate. Wall-E must get the box into the new brown box so he can go back home and rest.*): Fala sobre os objetivos e a narrativa do jogo.
- Instruções do jogo (*Use the arrow buttons to move and get the crate to the destination.*): Menciona os controles do jogo para controlar o personagem principal e o objetivo do jogo.
- Mensagem ao colocar a caixa no destino (*You won the level*): Mensagem de vitórias ao chegar no primeiro objetivo.
- Mensagem ao colocar a caixa no destino (*You won*): Mensagem de vitórias ao chegar no último objetivo.
- Steps: Representa a quantidade de passos que o jogador dá para conseguir chegar ao objetivo.

Metamensagem:

Na visão do designer, o usuário é alguém que conhece e gosta do filme Wall-E e também de uma história e objetivo mais detalhados mesmo que para um jogo simples. O jogo utiliza comandos simples para que o usuário não tenha dificuldade no início e oferece mensagens de feedback para o usuário.

SIGNOS ESTÁTICOS:

- Caixa com polígono vermelho claro: Representa o último objetivo do jogo
- Caixa com polígono vermelho escuro: Representa o primeiro objetivo do jogo
- Robô: Representa o personagem principal e controlado pelo jogador.
- Quadrados azuis: Representam os limites do campo da fase, onde o personagem não pode passar por cima.
- Ladrilho preto: Representa o local onde o robô pode se locomover livremente.
- Caixa: Representa o objeto que o robô deve locomover e levá-lo ao objetivo.
- Parte Branca: Representa o fundo do jogo

Metamensagem:

O jogo apresenta uma interface limpa com objetos que apenas representam formas de quadrado, o que pode tornar o jogo um pouco repetitivo visualmente.

SIGNOS DINÂMICOS:

- Robô: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Caixa: O movimento das caixas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.
- Número ao lado do *Steps*: A mudança do número de *Steps* representa a quantidade de ações de movimento realizadas pelo personagem principal.

Metamensagem:

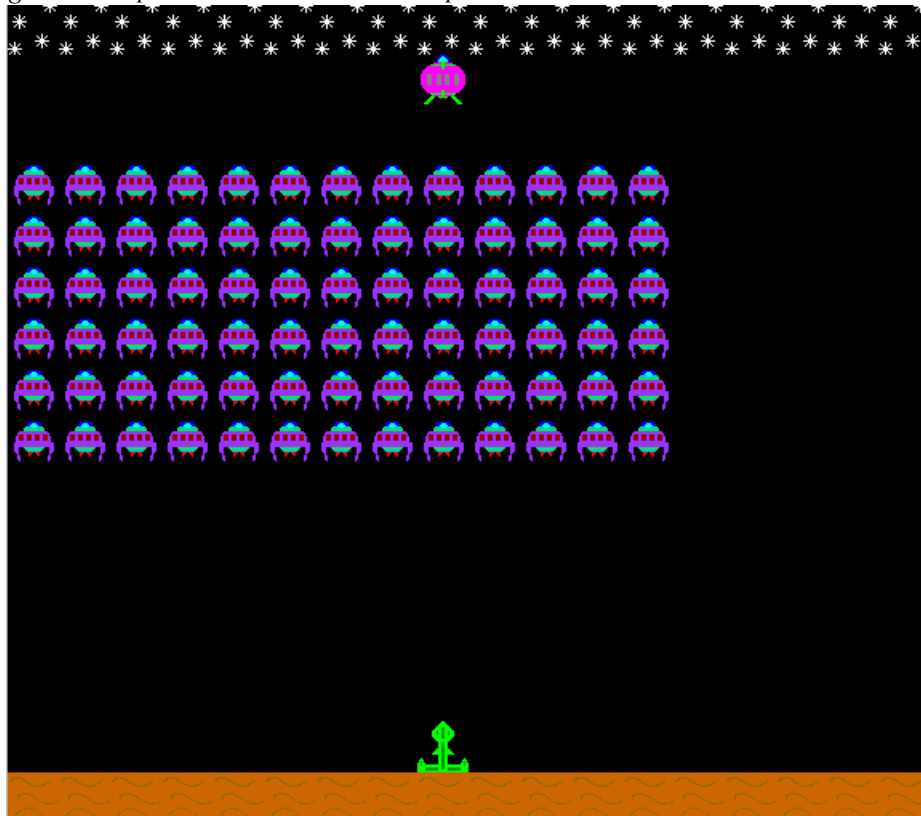
O jogo apresenta um objetivo simples com comandos simples, o que apresenta a ideia de um usuário que gosta de jogos rápidos e sem dificuldades de aprendizagem sobre o jogo. Além disso, na visão do designer, o usuário é alguém que gosta de se desafiar, pois ele pode sempre ver a quantidade de passos já dados no jogo e tentar melhorar essa pontuação cada vez mais.

COMPARAÇÃO:

Em todos os signos podemos ver a ideia de um jogo simples, no entanto, apenas nos signos metalinguísticos pode-se observar que o usuário é alguém que gosta de desafiar a si mesmo, fazendo com que ele possa repetir a fase para tentar completá-la com o mínimo de passos possíveis.

A02 - SPACE INVADERS

Figura 14. Space Invaders Desenvolvido por A02



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*You are a laser cannon. The aliens are invading and Earth's only hope is you. You must shoot the mothership to stop the aliens before they invade.*): Apresenta uma narrativa sobre o jogo e fala qual é o objetivo.
- Instruções do jogo (*Use the arrow keys (left and right) to move the laser cannon. Press the space bar to shoot.*): Diz quais são os controles do jogo e o que eles fazem.
- Mensagem ao ser atingido (*Your cannon was destroyed! Game over!*): Representa a mensagem de derrota.

Metamensagem:

O designer se baseia em um usuário aprecia atos heroicos como proteger a Terra, mesmo que seja com um objetivo simples. Além disso, o jogo não exige comandos complexos para interagir.

SIGNOS ESTÁTICOS:

- Canhão: Representa o personagem principal e controlável do jogo.
- Quadrado Listrado: Representa o chão onde o canhão pode se locomover livremente.
- Naves azuis: Naves de ataque que representam perigo para o canhão.
- Nave lilás: Representa a nave-mãe e objetivo do jogo.
- Estrelas: Representa o céu e o limite superior do jogo.

Metamensagem:

O jogo se baseia em elementos que representam alienígenas e objetos espaciais, o que está de acordo com um contexto de invasão ao planeta, ou seja, o designer vê o usuário como alguém que goste do contexto espacial. Os elementos são bastante coloridos, o que passa a ideia de um jogo divertido e não sério.

SIGNOS DINÂMICOS:

- Canhão: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Nave azul: O movimento da nave azul representa um aumento na dificuldade para destruí-la.
- Nave lilás: O movimento da nave lilás representa um aumento na dificuldade para destruí-la.
- Mísseis: O movimento dos mísseis representa ação causal que pode destruir o canhão
- Lazer: O movimento do laser representa ação causal para destruir as naves inimigas

Metamensagem:

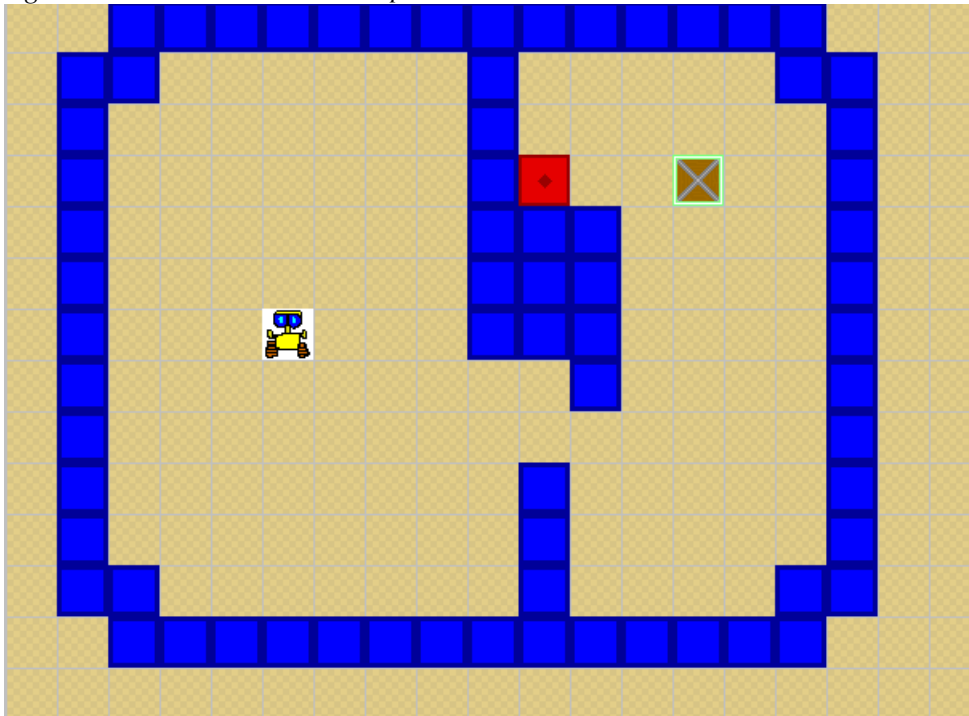
O designer utiliza diversos elementos que estão de acordo com a vida real, como mísseis e lasers como objetos de destruição, o que dá uma ideia de que o usuário gosta de jogos em que o objetivo é a destruição. Além disso, ele exige um pouco de rapidez e destreza por parte do usuário, pois o personagem principal já começa em uma área de perigo, além de apresentar diversos obstáculos que são perigosos ao personagem principal, o que torna o jogo mais difícil.

COMPARAÇÃO:

Comparando os signos, pode-se observar a concordância entre todos os três tipos quando se diz respeito ao contexto de invasão alienígena e que o jogo é divertido. Não foi observada nenhuma divergência entre os signos.

A03 - SOKOBAN

Figura 15. Sokoban Desenvolvido por A03



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*If you make the dude push the box in the destiny you get to the next level.*): Menciona o objetivo do jogo.
- Instruções do jogo (*By pressing the button up the dude moves up. By pressing the button down the dude moves down. By pushing the button left the dude moves to the left. By pushing the button right the dude moves to the right. You move the box by pushing it*): Mostra os controles do jogo.
- Mensagem ao atingir o objetivo (*Lets go*): Mensagem de vitória.

Metamensagem:

O designer se baseia em um usuário que gosta de jogos rápidos e simples que não exigem histórias ou comandos complexos.

SIGNOS ESTÁTICOS:

- Robô: Representa o personagem principal e controlável do jogo
- Quadrados azuis: Representam a parede ou limite de onde o personagem pode se locomover.
- Ladrilho: Representa a área onde o personagem pode se locomover livremente.
- Caixa: Objeto que deve ser levado ao objetivo
- Quadrado vermelho: Objetivo do jogo, onde a caixa deve ser levada.

Metamensagem:

O designer se baseia em um usuário que gosta do tema do Wall-E e de jogos que não apresentam uma interface detalhada, pois o jogo é feito apenas com formas quadrangulares, o que pode deixar o jogo repetitivo visualmente. Além disso, o jogo apresenta um nível de dificuldade baixo, pois é necessário colocar apenas uma caixa no lugar correto, o que reforça a idéia de um jogo simples.

SIGNOS DINÂMICOS:

- Robô: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Caixa: O movimento das caixas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.

Metamensagem:

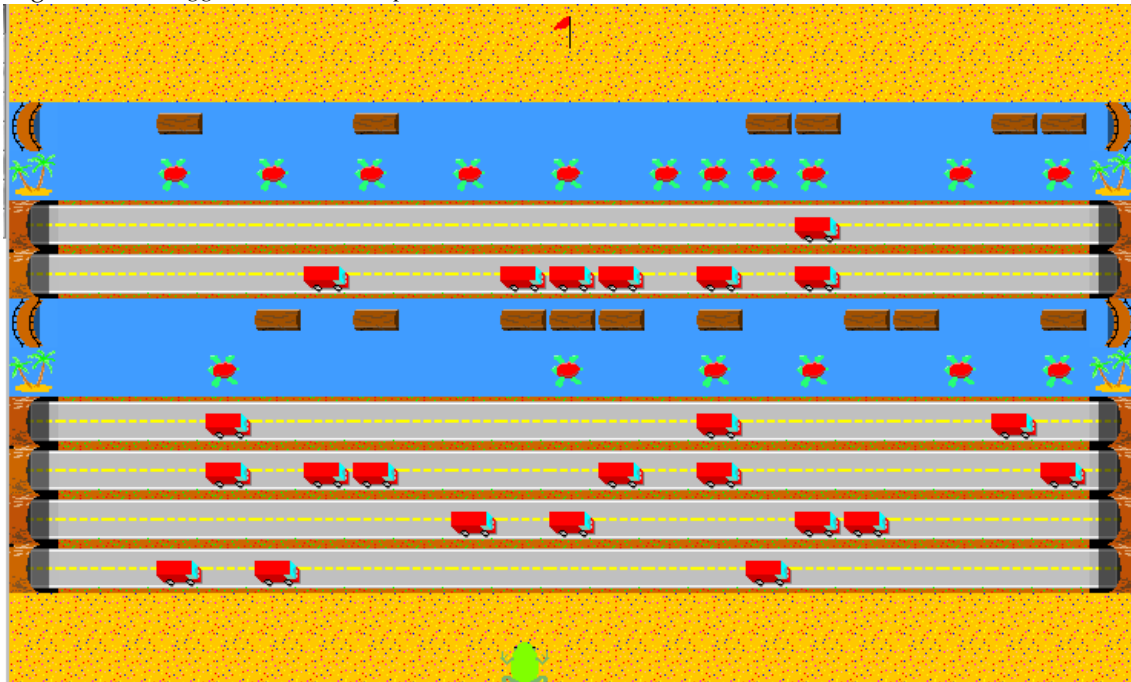
O designer baseia o jogo em um usuário que gosta de jogos com objetivos simples e com poucas interações.

Comparação:

Os signos dinâmicos e estáticos apresentam um contexto do filme Wall-e que não foi citado nos signos metalinguísticos. Todos os signos passam a ideia de um jogo simples e rápido.

A04 - FROGGER

Figura 16. Frogger Desenvolvido por A04



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Traditional Frogger with an exciting turn!*): Menciona um nível novo em relação ao Frogger tradicional.
- Instruções do jogo (*Use the arrows to move! Be careful not to sink in the water. Don't get ran over by the trucks/logs and turtles! ENJOY*): Dá algumas instruções do jogo e fala sobre os perigos representados pelos caminhões, troncos e tartarugas.
- Mensagem ao cair na água (*I cannot swim*): Mensagem de morte ao cair na água.
- Mensagem ao chegar na bandeira (*I made it*): Mensagem de vitória ao chegar ao objetivo.
- Mensagem ao tentar subir na ponte/ilha/túnel (*You cannot cheat*): Mensagem que informa que o jogador está tentando trapacear.

Metamensagem:

Na visão do designer, o usuário é alguém que conhece e gosta do Frogger tradicional e que queria um nível mais difícil dele. Além disso, ele apresenta os perigos que o jogo pode conter antes de o usuário interagir, ou seja, se baseia em um usuário que não gosta de surpresas no jogo. O jogo também apresenta diversas mensagens de feedback ao serem realizadas determinadas ações pelo usuário.

SIGNOS ESTÁTICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idênticos aos do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

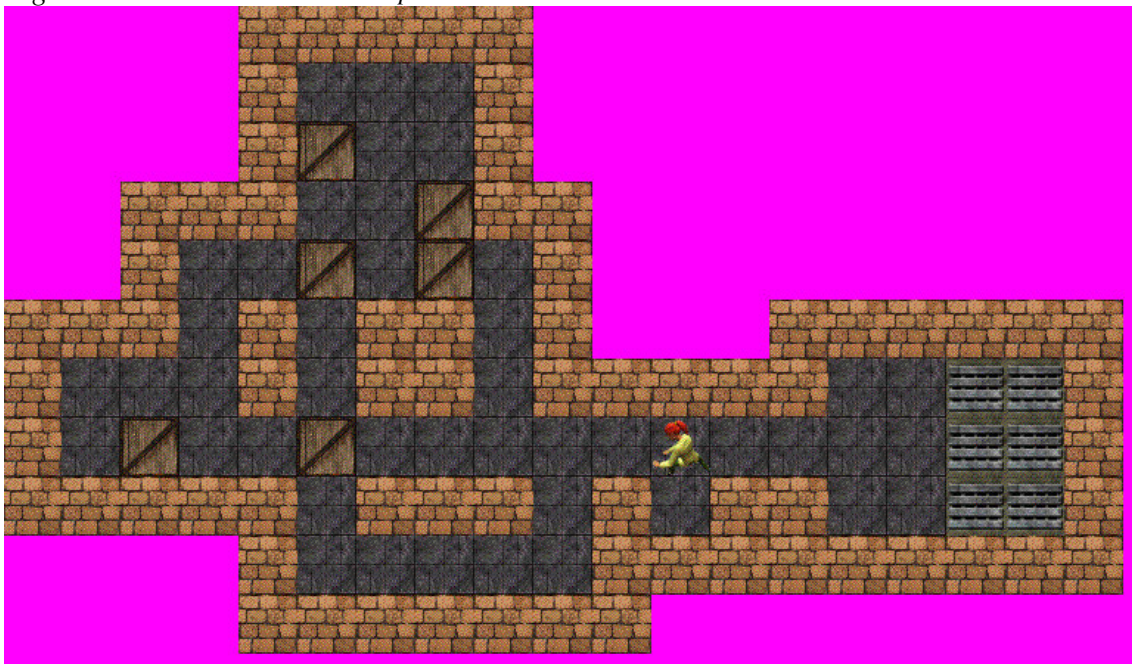
Idênticos aos do Frogger Original da seção 5.1.1.1

COMPARAÇÃO:

Na comparação entre os signos, um problema observado foi que nos signos metalinguísticos, o designer apresenta os signos tronco e tartaruga como elementos perigosos ao sapo. No entanto, como visto nos signos dinâmicos, estes elementos ajudam o sapo a atravessar o rio. O jogo também apresenta um bug nos caminhões, pois pela velocidade de alguns serem maiores que outros, se o jogador esperar alguns segundos para passar, a rua estará congestionada e impossível de se atravessar. Além disso, o usuário fala sobre um nível excitante além do normal, mas ao chegar na segunda bandeira, o sapo simplesmente é transportado para o início da mesma fase, resetando o jogo.

A04 - SOKOBAN

Figura 17. Sokoban Desenvolvido por A04



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Tradition Sokoban game.*): Menciona que é apenas o jogo tradicional, sem modificações.
- Instruções do jogo (*Use the arrows to move Sokoban, and take them to the grey tiles*): Menciona os controles do jogo e o objetivo.

Metamensagem:

Na visão do designer, o usuário é alguém que conhece e gosta do Sokoban original, sem modificações ou desafios maiores. O jogo oferece uma jogabilidade simples assim como o objetivo, que é o mesmo do sokoban original.

SIGNOS ESTÁTICOS:

- Tijolos: Representam a parede e o limite de onde o jogador pode andar
- Ladrilho: Representa a área em que o personagem pode se locomover livremente.
- Boneca: Representa o personagem principal e controlável.
- Plataforma de metal: Representa o objetivo do jogo onde as caixas devem ser deixadas.
- Caixas: Representam os objetos que devem ser levados ao objetivo do jogo.
- Área rosa: Lugar não jogável, serve apenas como fundo do jogo.

Metamensagem:

Idênticos aos do Sokoban Original da seção 5.1.1.3

SIGNOS DINÂMICOS:

Idênticos aos do Sokoban Original da seção 5.1.1.3

Metamensagem:

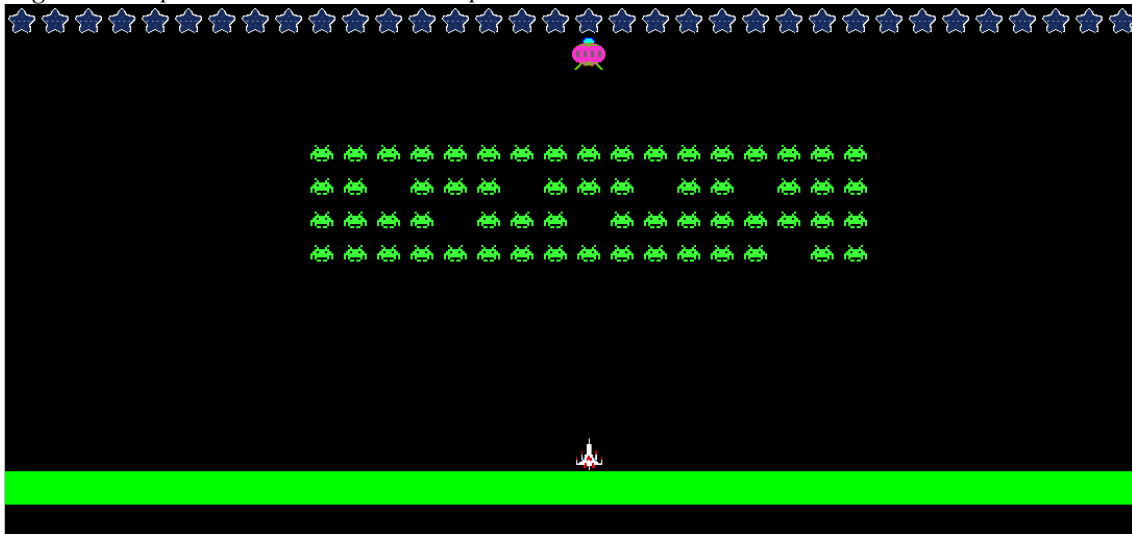
Idênticos aos do Sokoban Original da seção 5.1.1.3

COMPARAÇÃO:

Comparando os signos, observa-se que o designer teve a intenção de produzir um jogo calmo, sem perigos imediatos ou surpresas. Além disso, todos estão de acordo com a simplicidade vista nos metalinguísticos. O jogo exige que o usuário pense um pouco antes de realizar uma jogada, pois um erro pode impossibilitar a vitória do jogador.

A04 - SPACE INVADERS

Figura 18. Space Invaders Desenvolvido por A04



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Kill the Mothership to win the game and don't get shoot by the aliens. If you do the game will be over , don't let the aliens reach the floor either!*): Menciona o objetivo do jogo e o perigo sobre ser atingido pelos alienígenas ou deixá-los alcançar o chão.
- Instruções do jogo (*Use the arrow keys to move your ship and the space bar to fire lasers.*): Menciona os controles para mover e atirar com o personagem principal.
- Mensagem ao ser atingido por míssil inimigo (*Your ship was destroyed! Game over!*): Representa a morte do jogador.
- Mensagem ao pouso de um alienígena (*Game Over*): Representa a derrota do jogador ao deixar que os alienígenas invadam a terra.

Metamensagem:

Na visão do designer, o usuário é alguém que não gosta de descobrir os perigos por si só, pois ele já deixa claro todos os perigos que o jogo possui e que também gosta de uma narração rápida sobre o jogo. O sistema possui uma jogabilidade simples de como controlar o personagem principal. O jogo também apresenta mensagens de feedback ao serem realizadas determinadas ações do usuário e um contexto de invasão espacial.

SIGNOS ESTÁTICOS:

- Nave: Representa o personagem principal e controlável do jogo.
- Listra Amarela: Representa o chão onde o canhão pode se locomover livremente.
- Alienígenas: Seres que representam perigo para o canhão.
- Nave lilás: Representa a nave-mãe e objetivo do jogo.
- Estrelas: Representa o céu e o limite superior do jogo.

Metamensagem:

O jogo se baseia em elementos que representam alienígenas e objetos espaciais, o que está de acordo com um contexto de invasão ao planeta, ou seja, o designer vê o usuário como

alguém que goste do contexto espacial. Os elementos são bastante coloridos, o que passa a ideia de um jogo divertido e não sério.

SIGNOS DINÂMICOS:

- Nave: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Alienígena: O movimento do alienígena representa um aumento na dificuldade para destruí-lo.
- Nave lilás: O movimento da nave lilás representa um aumento na dificuldade para destruí-la.
- Mísseis: O movimento dos mísseis representa ação causal que pode destruir o canhão
- Laser: O movimento do laser representa ação causal para destruir as naves inimigas

Metamensagem:

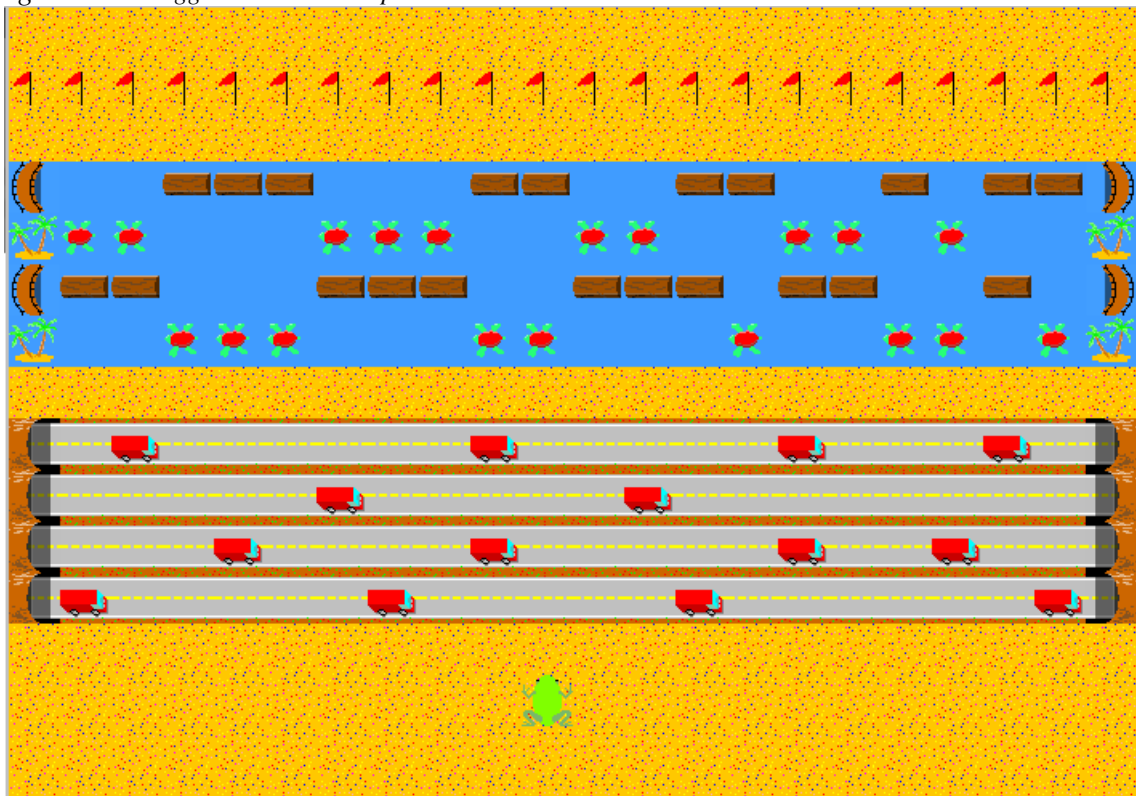
O designer utiliza diversos elementos que estão de acordo com a vida real, como mísseis e lasers como objetos de destruição, o que dá uma ideia de que o usuário gosta de jogos em que o objetivo é a destruição. Além disso, ele exige um pouco de rapidez e destreza por parte do usuário, pois o personagem principal já começa em uma área de perigo, onde as naves inimigas podem matá-lo, tornando o jogo mais difícil.

COMPARAÇÃO:

Todos os signos estão de acordo com o contexto citado nos signos metalinguísticos, que é o espacial. Eles também estão de acordo no quesito de simplicidade do jogo. Não foi observada nenhuma divergência entre os signos.

A05 - FROGGER

Figura 19. Frogger Desenvolvido por A05



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*You are a frog and you have to get to the other side of the road and river.*): Mostra uma pequena contextualização do jogo e como chegar ao objetivo.
- Instruções do jogo (*To start playing press the green arrow. To move use the arrow keys*): Menciona os controles necessários para jogar.

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos rápidos e simples, pois já deixa claro quem é o usuário dentro do jogo e o objetivo. Ele também apresenta a forma de dar início ao jogo e como se movimentar, o que evita que o usuário tenha que explorar bem o jogo para saber o que fazer.

SIGNOS ESTÁTICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idênticos aos do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

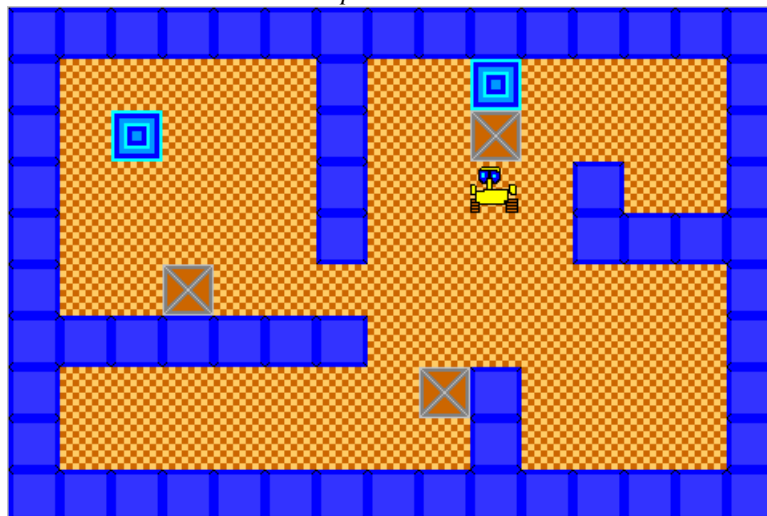
Idênticos aos do Frogger Original da seção 5.1.1.1

COMPARAÇÃO:

Comparando os signos, podemos ver que o jogo oferece simplicidade e facilidade de entendimento sobre o jogo, como dito nos signos metalinguísticos e apresentado pelos dinâmicos e estáticos. Não foi observada nenhuma divergência entre os signos.

A05 - SOKOBAN

Figura 20. Sokoban Desenvolvido por A05



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Your object is to move the crate on top of the goal (blue colored block)*): Menciona o objetivo do jogo.
- Instruções do jogo (*Use the arrow keys to move sokoban. Sokoban will push the crates*): Menciona os controles para controlar o personagem.

Metamensagem:

O designer apresenta um objetivo simples usando comandos simples, o que dá a ideia de que o usuário é alguém que gosta de jogos rápidos e práticos.

SIGNOS ESTÁTICOS:

- Robô: Representa o personagem principal e controlável do jogo
- Quadrados azuis: Representam a parede ou limite de onde o personagem pode se locomover.
- Ladrilho: Representa a área onde o personagem pode se locomover livremente.
- Caixa: Objeto que deve ser levado ao objetivo
- Quadrado preenchido com listras: Objetivo do jogo, onde a caixa deve ser levada.

Metamensagem:

O designer se baseia em um usuário que gosta do tema do Wall-E e de jogos que não apresentam uma interface detalhada, pois o jogo é feito apenas de formas quadrangulares, o que pode deixar o jogo repetitivo visualmente.

SIGNOS DINÂMICOS:

- Robô: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Caixa: O movimento das caixas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.

Metamensagem:

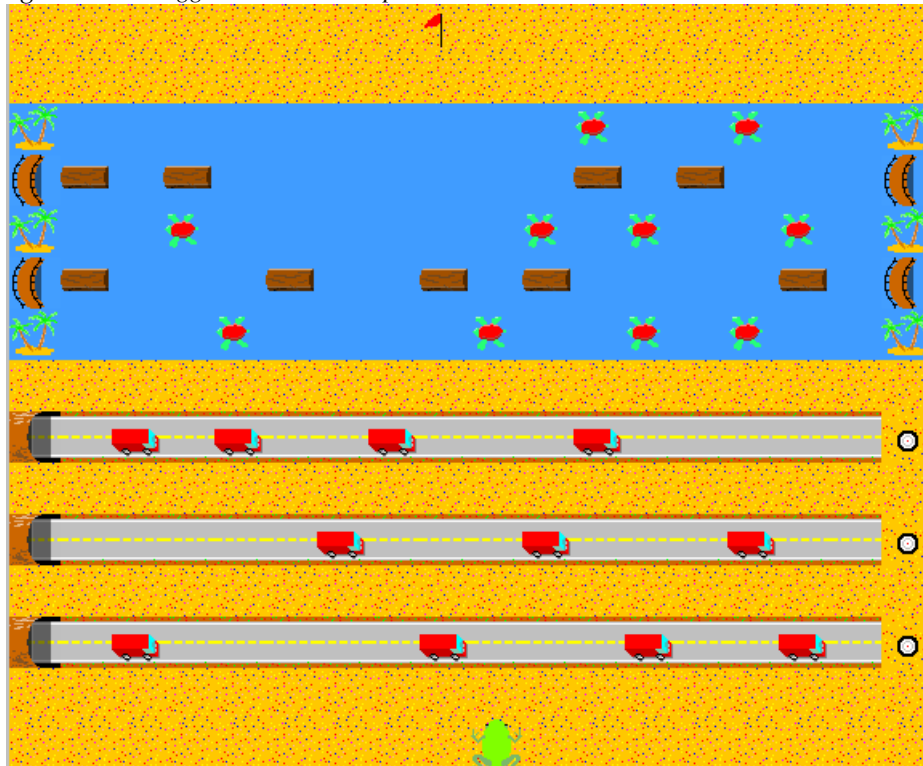
O designer baseia o jogo em um usuário que gosta de jogos com objetivos simples e com poucas interações. Mas que necessita pensar bastante antes de realizar uma jogada, pois em determinadas localizações é impossível mexer uma caixa.

COMPARAÇÃO:

Comparando os signos, observa-se que o designer, nos signos metalinguísticos, refere-se ao personagem principal como Sokoban (nome do jogo). Entretanto, quando ele usa o tema do Wall-E, como visto nos signos estáticos, este deveria ser o nome do personagem. Além disso, o jogo possui um bug, visto durante a inspeção dos signos dinâmicos, pois ele ganha ao colocar apenas uma caixa no lugar determinado, o que deveria ocorrer apenas quando ele colocasse todas as caixas nos lugares corretos. Ambos os signos estão de acordo com a simplicidade do jogo. Além disso, o jogo exige que o jogador pense antes de realizar uma jogada, pois um passo errado pode impossibilitar a vitória do jogador.

A06 - Frogger

Figura 21. Frogger Desenvolvido por A06



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*This is a game that will make you think and hopefully it will be fun*): Fala sobre o objetivo da criação do jogo.
- Instruções do jogo (*Use the arrows on your keyboard to move the mouse. Your objective is to move the cheese into his house. Have Fun!*): Menciona os controles necessários para mover o personagem principal e o objetivo do jogo.
- Mensagem ao chegar na última bandeira (*You won!!*): Mensagem que representa a vitória do jogador.

Metamensagem:

O designer se baseia em um usuário que gosta de jogos em que ele deve pensar antes de agir e mesmo assim se divertir. Ele também apresenta um objetivo e comandos simples. Além disso, ele se baseia em um usuário que gosta de receber um feedback de vitória ao concluir o objetivo do jogo.

SIGNOS ESTÁTICOS:

- Sapo: Representa o personagem principal do jogo
- Grama: Representa uma área segura para o sapo
- Estrada: Representa uma área de perigo para o sapo
- Bandeira: Representa o ponto de conclusão da fase
- Túnel: Representam os pontos de onde os caminhões aparecem no jogo
- Círculos Brancos: Representam o ponto onde os caminhões desaparecem do jogo.
- Ilha: Representam os pontos de onde as tartarugas aparecem e somem do jogo

- Ponte: Representam os pontos de onde os troncos aparecem e somem do jogo
- Água: Representa uma área de perigo para o sapo
- Tronco: Representa objeto que ajuda o sapo a passar pela água sem morrer
- Tartaruga: Representa objeto que ajuda o sapo a passar pela água sem morrer
- Caminhão: Representa um obstáculo que pode matar o sapo

Metamensagem:

Idêntica ao do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

- Sapo: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo
- Tronco: O movimento do tronco representa ação causal que pode matar o sapo
- Tartaruga: O movimento da tartaruga representa ação causal que pode matar o sapo
- Caminhão: O movimento do caminhão representa ação causal que pode matar o sapo
- Fumaça preta: A animação da fumaça representa a exclusão do caminhão do mapa de *tiles*

Metamensagem:

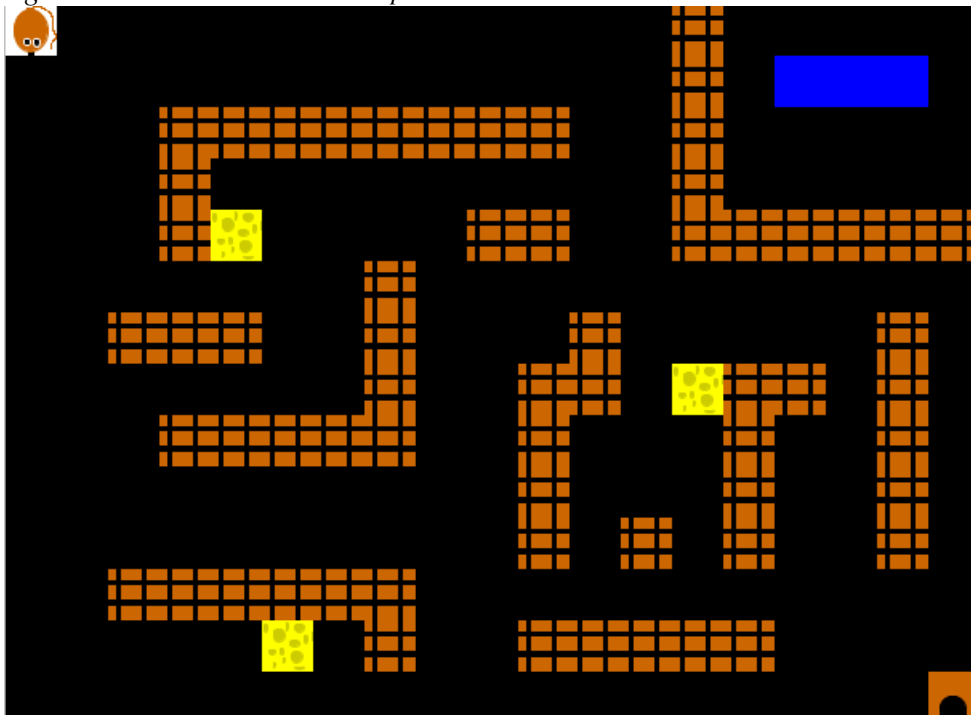
Idênticos aos do Frogger Original da seção 5.1.1.1

COMPARAÇÃO:

Um problema observado durante a comparação entre os signos foi que, nos signos metalinguísticos, o designer age como se o personagem principal fosse um rato que deveria chegar ao queijo, quando, na verdade, é um sapo que deve chegar à bandeira, como visto nos signos estáticos e dinâmicos. A seção de signos estáticos traz o “Círculo Branco” que parece ser um bug na interface, pois o jogo apresenta uma temática realística e esse “círculo branco” não faz sentido no meio do contexto. Nos signos metalinguísticos também é dito que o usuário deveria precisar pensar bastante quando, na verdade, o jogo exige mais paciência que estratégia.

A06 - Sokoban

Figura 22. Sokoban Desenvolvido por A06



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*This is a game that will make you think and hopefully it will be fun*): Fala sobre o objetivo da criação do jogo.
- Instruções do jogo (*Use the arrows on your keyboard to move the mouse. Your objective is to move the cheese into his house. Have Fun!*): Menciona os controles necessários para mover o personagem principal e o objetivo do jogo.

Metamensagem:

Na visão do designer, o jogo foi criado para usuários que gostam de jogos divertidos que tenham de utilizar a inteligência para cumprir o objetivo exigido no jogo. Além disso, o usuário é alguém que gosta de jogos com comandos simples para movimentação do personagem.

SIGNOS ESTÁTICOS:

- Rato: Representa o personagem principal do jogo.
- Queijo: Representa o objeto que deve ser colocado dentro da casa.
- Ladrilhos Vermelhos: Parede onde o sapo não pode se locomover e nem colocar os queijos sobre.
- Casinha: Objetivo do jogo, onde os queijos devem ser levados.
- Água: Apenas um enfeite do jogo.
- Área preta: Local onde o rato pode se locomover e arrastar os queijos livremente

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com tema e interface divertidos. Além disso, o jogo faz alusão à realidade, pois o rato tem que levar o queijo para dentro de seu esconderijo, dando uma ideia de que o usuário gosta de temas realísticos.

SIGNOS DINÂMICOS:

- Rato: Representa o personagem principal do jogo.
- Queijo: Representa o objeto que deve ser colocado dentro da casa.

Metamensagem:

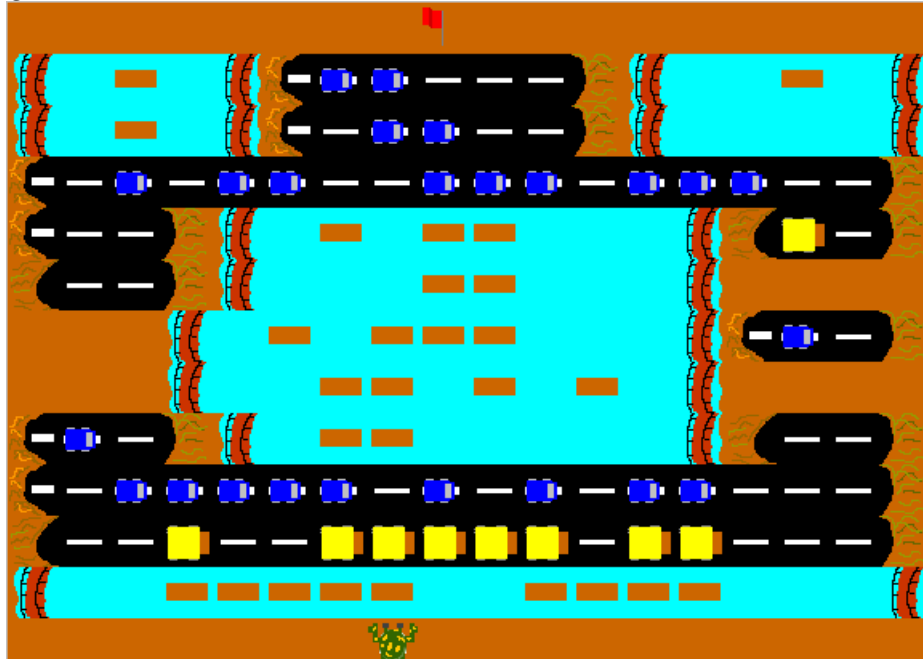
Na visão do designer, o usuário é alguém que gosta de jogos com objetivos simples e com poucas interações, mas que precisem de uma estratégia mínima para poder vencer.

COMPARAÇÃO:

Realizando a comparação, observamos que todos os tipos de signos apresentam um jogo que deve ser divertido e ao mesmo tempo exige uma estratégia para chegar ao objetivo. Não foi observado nenhuma divergência entre os signos.

A07 - FROGGER

Figura 23. Frogger Desenvolvido por A07



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*This is a frogger adaption made by me, everything is original, as i drew and gave directions to all of the agents.*): Menciona uma inovação em relação ao jogo original.
- Instruções do jogo (*Use the arrows to move the frog and get past all of the obstacles, remember, if you cheat, you shall die. Cars and trucks shall maul you and kill you fiercly, you cant swim. remember that.*): Menciona os controles para movimentar o personagem principal e o perigo de trapacear no jogo como, também, sobre os perigos representados pela água e pelos caminhões.
- Mensagem ao cair na água (*I CANT SWIN*): Representa a morte do personagem.
- Mensagem ao chegar no primeiro objetivo (*YAY I did it, thanks man*): Representa a chegada do personagem ao primeiro objetivo.
- Mensagem ao chegar no último objetivo (*You Beat the Game!! Thank for playing*): Representa a finalização do jogo e vitória do jogador.

Metamensagem:

Na visão do designer, o usuário é alguém que conhece e gosta do jogo original do Frogger, mas que vai gostar mais ainda do jogo feito por ele. O jogo apresenta comandos simples e obstáculos que podem matar o personagem principal, o que dá a ideia de que o usuário gosta de jogos mais desafiadores. O designer também vê o usuário como alguém que gosta de saber a funcionalidade dos elementos do jogo antes de iniciá-lo. O jogo apresenta diversas mensagens de feedback divertidas ao ocorrer determinada ação.

SIGNOS ESTÁTICOS:

- Sapo: Representa o personagem principal do jogo
- Grama: Representa uma área segura para o sapo

- Estrada: Representa uma área de perigo para o sapo
- Bandeira: Representa o ponto de conclusão da fase
- Traços como continuação da estrada: Representam os pontos de onde os caminhões e os carros aparecem e somem do jogo
- Ilha: Representam os pontos de onde as tartarugas aparecem e somem do jogo
- Ponte: Representam os pontos de onde os troncos aparecem e somem do jogo
- Água: Representa uma área de perigo para o sapo
- Tronco: Representa objeto que ajuda o sapo a passar pela água sem morrer
- Tartaruga: Representa objeto que ajuda o sapo a passar pela água sem morrer
- Caminhão e carro: Representam obstáculos que podem matar o sapo

Metamensagem:

O designer se baseia em signos que são conhecidos da rotina do usuário, como automóveis, animais, pontes, etc. e que gostam de cenários coloridos. O usuário é alguém que gosta de um tempo para analisar o cenário antes de entrar em zonas perigosas, pois o sapo, que é o personagem principal, começa em uma larga área segura. O usuário pode entender a bandeira como um símbolo de vitória, pois desde o tempo dos conquistadores, era um símbolo de vitória quando eles conseguiam alcançar seus objetivos.

SIGNOS DINÂMICOS:

- Sapo: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo
- Tronco: O movimento do tronco representa ação causal que pode matar o sapo
- Tartaruga: O movimento da tartaruga representa ação causal que pode matar o sapo
- Caminhão e carro: O movimento do caminhão representa ação causal que pode matar o sapo

Metamensagem:

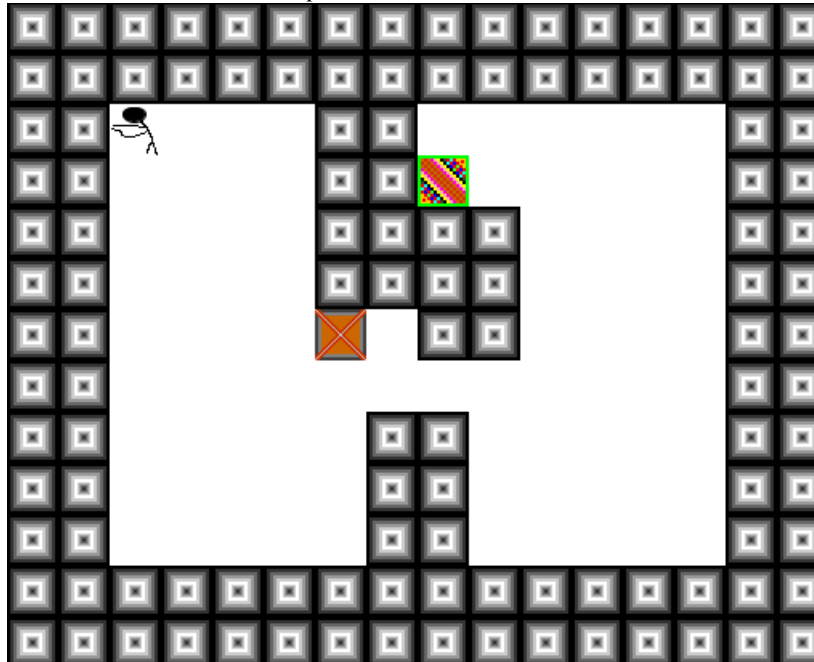
Na visão do designer, o usuário é alguém que gosta de ter tempo para pensar e poder voltar os obstáculos, pois o sapo pode andar para trás e voltar nos obstáculos. O jogo apresenta um nível mais difícil onde os caminhões, tartarugas e troncos são mais rápidos e em maior quantidade, aumentando as chances de cair na água ou ser atropelado, o que leva a pensar que o usuário gosta de níveis mais desafiadores.

COMPARAÇÃO:

Durante a comparação dos signos, foi observado que os três tipos de signos dão a ideia de um jogo simples. Além disso, o designer também fala sobre a funcionalidade de cada tipo de elemento do jogo, o que é visto nos signos estáticos e dinâmicos, que apresentam elementos realistas com as mesmas funções da vida real, facilitando o entendimento sobre o jogo.

A07 - Sokoban

Figura 24. Sokoban Desenvolvido por A07



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Put The Box Into Its Destination!*): Menciona o objetivo do jogo.
- Instruções do jogo (*Use the up, right, down, and left arrows to move!*): Menciona os controles para mover o personagem principal.
- Mensagem ao colocar as caixas nos objetivos secundários (*Nice! Lets go to the next level!*): Representa vitórias do personagem nas primeiras fases.
- Mensagem ao colocar as caixas no objetivo final (*Well done!! You beat the game!*): Mensagem de vitória e finalização do jogo.

Metamensagem:

Na visão do designer, o usuário é alguém que não gosta de perder muito tempo para iniciar o jogo e que prefere descrição e comandos simples. O jogo oferece mensagens de feedback divertidas para o usuário ao ocorrer determinadas ações.

SIGNOS ESTÁTICOS:

- Boneco palito: Representa o personagem principal do jogo.
- Caixa vermelha: Representa os objetos que devem ser levados ao objetivo.
- Quadrado verde com um X: Representa o objetivo final do jogo e onde as caixas devem ser colocadas.
- Quadrado preenchido com listras: Representam os objetivos iniciais do jogo e onde as caixas vermelhas devem ser colocadas.
- Caixas preenchidas com branco e cinza: Representa a área onde o personagem principal não pode andar e colocar as caixas sobre.
- Área branca: Representa onde o personagem pode andar e arrastar caixas livremente

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de uma interface simples, porém, divertida, com personagem principal engraçado. No entanto, todas as fases oferecem os mesmos tipos de elementos, que acabam sendo cansativos visualmente.

SIGNOS DINÂMICOS:

- Boneco palito: Representa o personagem principal do jogo.
- Caixa vermelha: Representa os objetos que devem ser levados ao objetivo.

Metamensagem:

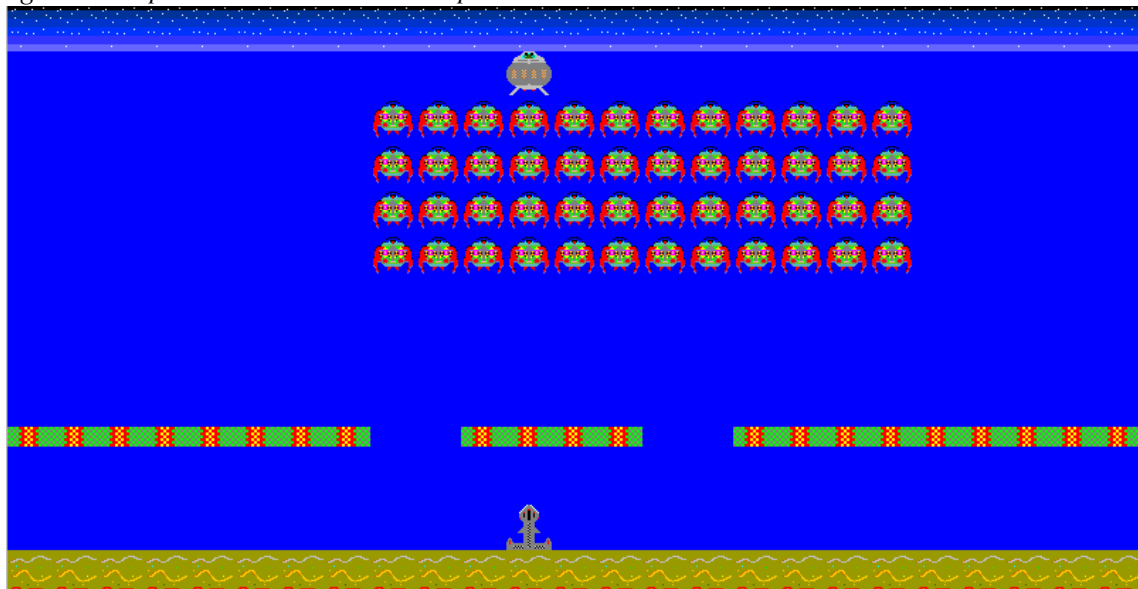
Na visão do designer, o usuário é alguém que gosta de jogos com objetivos simples e com poucas interações.

COMPARAÇÃO:

No resultado da comparação entre os signos, pode-se observar que todos dão a ideia de um jogo simples e rápido. Além disso, vários elementos de todos os tipos de signos passam a ideia de um jogo divertido e engraçado. Não foram observadas divergências entre os signos.

A07 - SPACE INVADERS

Figura 25. Space Invaders Desenvolvido por A07



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Play space invaders!*): Induz o usuário a jogar.
- Instruções do jogo (*arrows to move, and space bar to shoot*): Menciona os controles para movimentar e atirar com o personagem principal.
- Mensagem ao ser atingido por um míssil (*You lost! Your only defense mechanism was destroyed!!!*): Representa a morte do personagem principal.
- Mensagem ao alienígena aterrissar (*The aliens HAVE landed! All humanity is lost!*): Representa a derrota do jogador.
- Mensagem ao destruir a nave-mãe (*Well done!!! You just saved Brazil! But quick, lets go to Tokyo were the aliens are taking over!*): Mensagem de vitória ao destruir a nave principal.

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com início rápido, sem muitas explicações e com comandos simples. O jogo oferece diversas mensagens de feedback divertidas ao ocorrerem determinadas ações no jogo. A mensagem ao destruir a nave-mãe muda o contexto do objetivo do jogo, que passa de destruir a nave-mãe para salvar determinados lugares do mundo.

SIGNOS ESTÁTICOS:

- Canhão: Representa o personagem principal e controlável do jogo.
- Listra colorida inferior: Representa o chão onde o canhão pode se locomover livremente.
- Naves pequenas: Seres que representam perigo para o canhão.
- Nave Maior: Representa a nave-mãe e objetivo do jogo.
- Estrelas: Representam o céu e o limite superior do jogo.
- Listra flutuante: Representa uma área de defesa

Metamensagem:

O jogo se baseia em elementos que representam alienígenas e objetos espaciais, o que está de acordo com um contexto de invasão ao planeta, ou seja, o designer vê o usuário como alguém que goste do conceito espacial. Os elementos são bastante coloridos, o que passa a ideia de um jogo divertido e não sério.

SIGNOS DINÂMICOS:

- Canhão: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Naves pequenas: O movimento das naves pequenas representa um aumento na dificuldade para destruí-las.
- Nave Maior: O movimento das naves grandes representa um aumento na dificuldade para destruí-las.
- Mísseis: O movimento dos mísseis representa ação causal que pode destruir o canhão
- Lazer: O movimento do laser representa ação causal para destruir as naves inimigas
- Listra flutuante: A mudança da listra flutuante representa a diminuição de sua vida

Metamensagem:

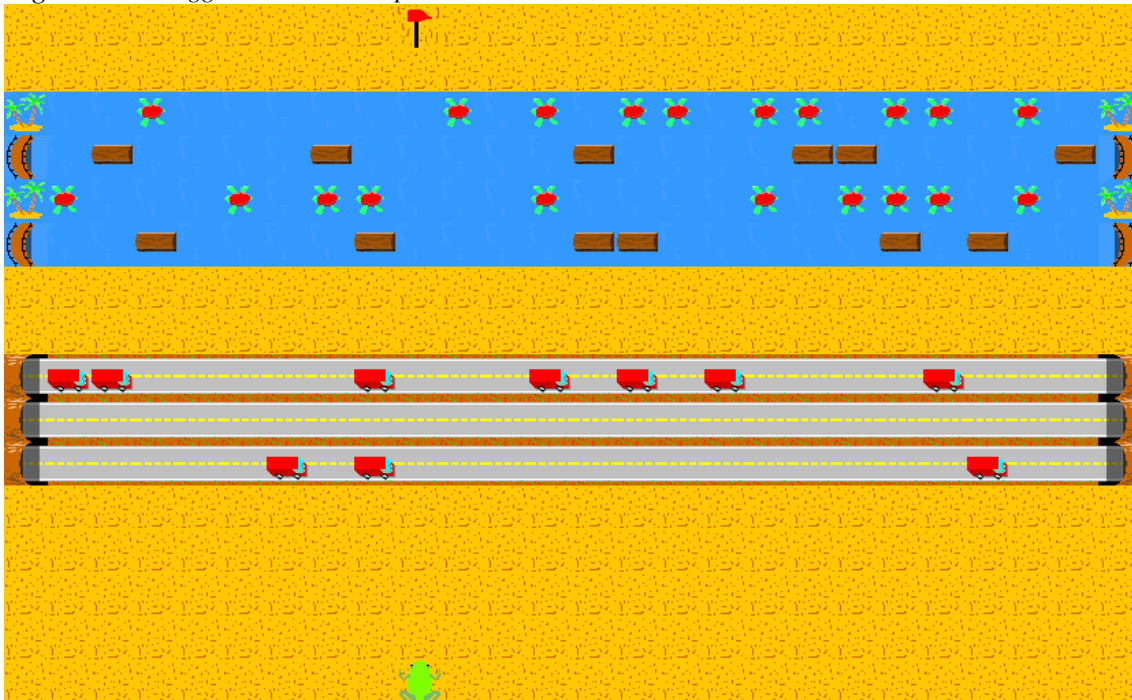
O designer utiliza diversos elementos que estão de acordo com a vida real, como mísseis e lasers como objetos de destruição, o que dá uma ideia de que o usuário gosta de jogos em que o objetivo é a destruição. Além disso, o jogo apresenta diversos obstáculos que representam perigo ao personagem principal, o que torna o jogo mais desafiador e uma área de proteção que exige um pouco mais de estratégia por parte do usuário.

COMPARAÇÃO:

Ao comparar os 3 tipos de signos, foi visto que todos tornam o jogo divertido de diferentes formas. Nos signos metalinguísticos, o designer muda o objetivo do jogo colocando o contexto de salvar países ao invés de apenas destruir a nave-mãe. Já os signos estáticos e dinâmicos apresentam diversos desafios e um contexto espacial.

A08 - FROGGER

Figura 26. Frogger Desenvolvido por A08



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Instruções do jogo (*Use the arrow keys to move the frog. get to the flag*): Menciona os controles para movimentar o personagem principal e o objetivo do jogo.
- Mensagem ao cair na água (*I can not swim*): Representa a morte do personagem principal.
- Mensagem ao chegar na bandeira (*I made it*): Representa a vitória do jogador ao cumprir um objetivo.

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com comandos simples e de início rápido, pois apresenta apenas as instruções necessárias para se mover. O jogo também oferece poucas mensagens de feedback, o que apresenta a ideia de um jogo que não necessita de feedbacks.

SIGNOS ESTÁTICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idêntica ao do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

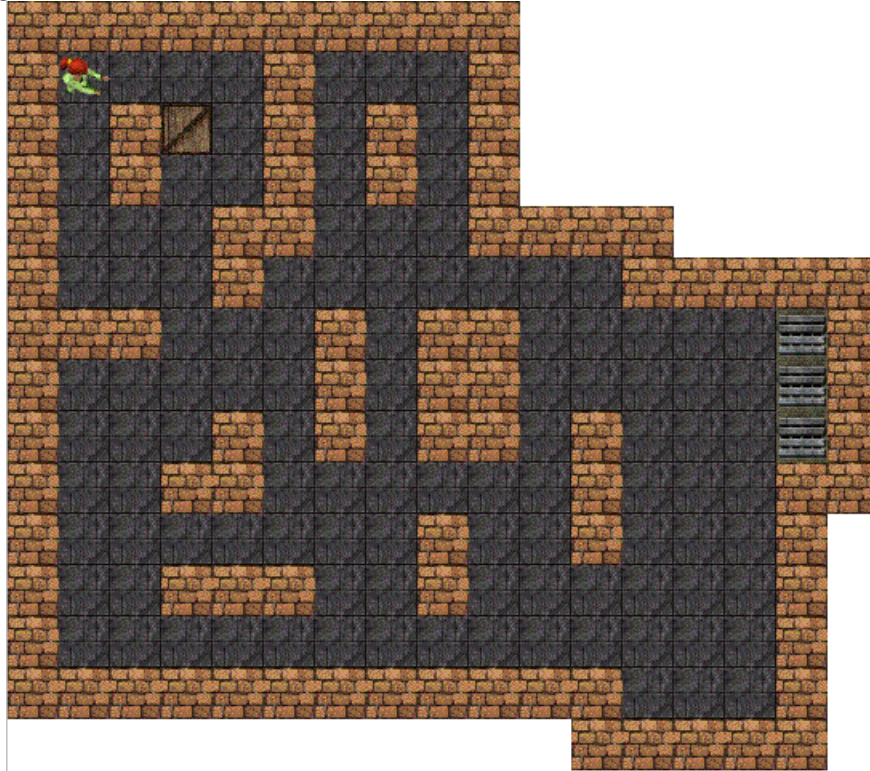
Idêntica ao do Frogger Original da seção 5.1.1.1

COMPARAÇÃO:

Durante a comparação foi visto que os signos metalinguísticos estavam incompletos se comparados com os estáticos e dinâmicos, que apresentam bem mais informações. No entanto, todos apresentam o jogo como sendo simples e rápido.

A08 - SOKOBAN

Figura 27. Sokoban Desenvolvido por A08



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Descrição do jogo (*Sokoban level 1 and 2*): Informa que o jogo possui 2 níveis
- Instruções do jogo (*In the first level you have to move the box and in the second level you have to move the box and put it on the red square*): Informa os objetivos do jogo e a diferença entre os níveis.
- Mensagem ao colocar a caixa no objetivo (*You won*): Representa a vitória do jogador

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com mais de uma fase e que cada uma tenha um objetivo diferente, mesmo que simples. Ele também assume que o usuário não gosta de ser atrapalhado com mensagens durante o jogo e oferece apenas um feedback de vitória.

SIGNOS ESTÁTICOS:

Idênticos aos do Sokoban Original da seção 5.1.1.3

Metamensagem:

Idêntica ao do Sokoban Original da seção 5.1.1.3

SIGNOS DINÂMICOS:

Idênticos aos do Sokoban Original da seção 5.1.1.3

Metamensagem:

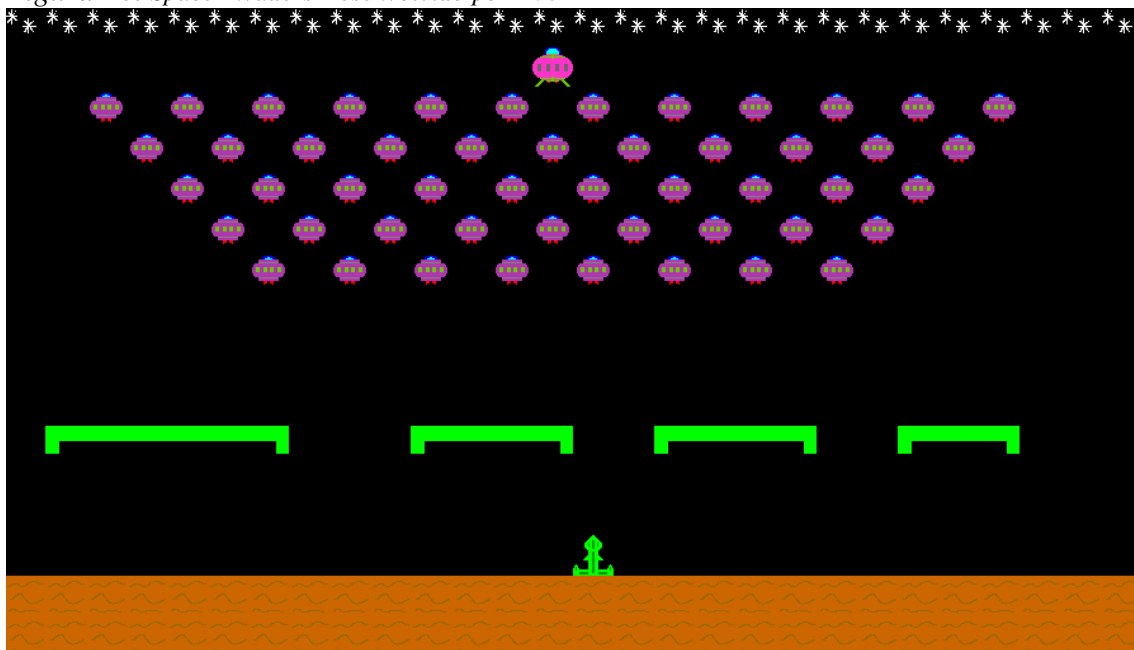
Idênticos aos do Sokoban Original da seção 5.1.1.3

COMPARAÇÃO:

Nos signos metalinguísticos, o designer faz parecer que o jogo possui dois objetivos diferentes, quando na verdade as duas fases possuem o mesmo objetivo, conclusão obtida apenas com os signos dinâmicos. No entanto, a simplicidade do jogo está presente nos três tipos de signos.

A08 - SPACE INVADERS

Figura 28. *Space Invaders* Desenvolvido por A08



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Mensagem ao destruir a primeira nave mãe (*Well done! Let's go to the next level*): Representa vitória do jogador e finalização do jogo.
- Mensagem ao alienígena aterrissar (*Alien fleet has landed! Game over*): Representa derrota do jogador.

Metamensagem:

Na visão do designer, o usuário é alguém que já conhece esse tipo de jogo e não precisa de nenhuma explicação. Ele também assume que o usuário não se interessa por nenhum tipo de narrativa ou descrição do jogo, apenas um feedback de vitória e apenas neste feedback ele informa que o jogo possui mais um nível.

SIGNOS ESTÁTICOS:

Idênticos aos do Space Invaders Original da seção 5.1.1.2

Metamensagem:

Idêntica ao do Space Invaders Original da seção 5.1.1.2

SIGNOS DINÂMICOS:

Idênticos aos do Space Invaders Original da seção 5.1.1.2

Metamensagem:

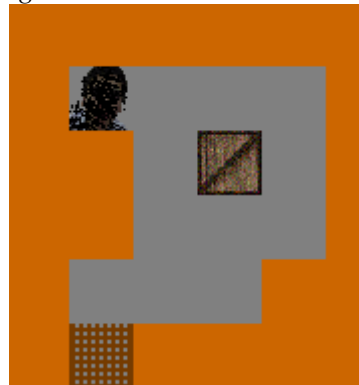
Idêntica ao do Space Invaders Original da seção 5.1.1.2

COMPARAÇÃO:

Durante a comparação foi observado que os signos metalinguísticos não oferecem muitas informações sobre o jogo, o que pode dificultar o entendimento e a jogabilidade. No entanto, os signos estáticos e dinâmicos apresentam elementos que facilitam o entendimento sobre o jogo, além de passar a ideia de um jogo simples e divertido.

A09 – SOKOBAN

Figura 29. Sokoban Desenvolvido por A09



Fonte: PoliFacets

SIGNOS METALINGUISTICOS:

- Descrição do jogo (*Sokoban with some wired controls. Will make anyone who has played normal sokoban, freakout!*): Informa que o jogo tem controles ligados e que se o jogador já jogou o sokoban original, irá gostar mais ainda deste.
- Instruções do jogo (*WASD to move*): Informa os comandos do jogo.
- Mensagem ao chegar no objetivo (*Game Over : You have won! You are part of the handful of people who have beaten this game!*): Apresenta a vitória do jogador.

Metamensagem:

Na visão do designer, o usuário é alguém que pode já ter jogado sokoban, mas que esse vai ser surpreendente. Além disso, ele também se baseia em usuário que já tem experiência com jogos, pois os comandos “AWS” não são comandos comuns de movimentação no teclado. O jogo só oferece uma mensagem de feedback, a de vitória do jogador.

SIGNOS ESTÁTICOS:

- Área cinza: Representa a área em que o personagem pode se locomover livremente.
- Rosto: Representa o personagem principal e controlável.
- Ladrilho vermelho: Representa o objetivo do jogo onde as caixas devem ser deixadas.
- Caixas: Representam os objetos que devem ser levados ao objetivo do jogo.
- Área laranja: Local não jogável, serve apenas como plano de fundo do jogo.

Metamensagem:

Na visão do designer, o jogo apresenta uma temática engraçada, provavelmente como gozação usando a foto de algum amigo. Além disso, o jogo oferece elementos simples e de fácil percepção.

SIGNOS DINÂMICOS:

- Rosto: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Caixa: O movimento das caixas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.

Metamensagem:

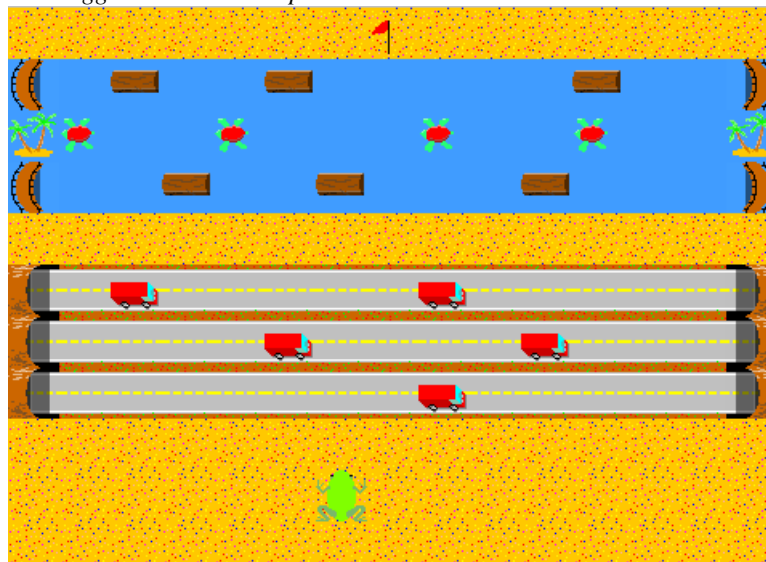
O designer assume que o usuário gosta de jogos simples e sem riscos, pois o jogo não apresenta nenhum elemento que represente perigo imediato ao personagem. Além disso, o usuário não tem a possibilidade de interagir muito com o cenário, o que dá a ideia de um jogo rápido.

COMPARAÇÃO:

Nos signos metalinguísticos, o designer apresenta o jogo com controles ligados, o que não faz sentido. Além disso, nos signos metalinguísticos ele deixa explícito que o jogo será diferente do original, mas a jogabilidade é exatamente a mesma, apenas alterando os comandos e os elementos de interface, como visto nos signos estáticos e dinâmicos.

A10 - FROGGER

Figura 30. Frogger Desenvolvido por A10



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

Não possui

SIGNOS ESTÁTICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idêntica ao do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

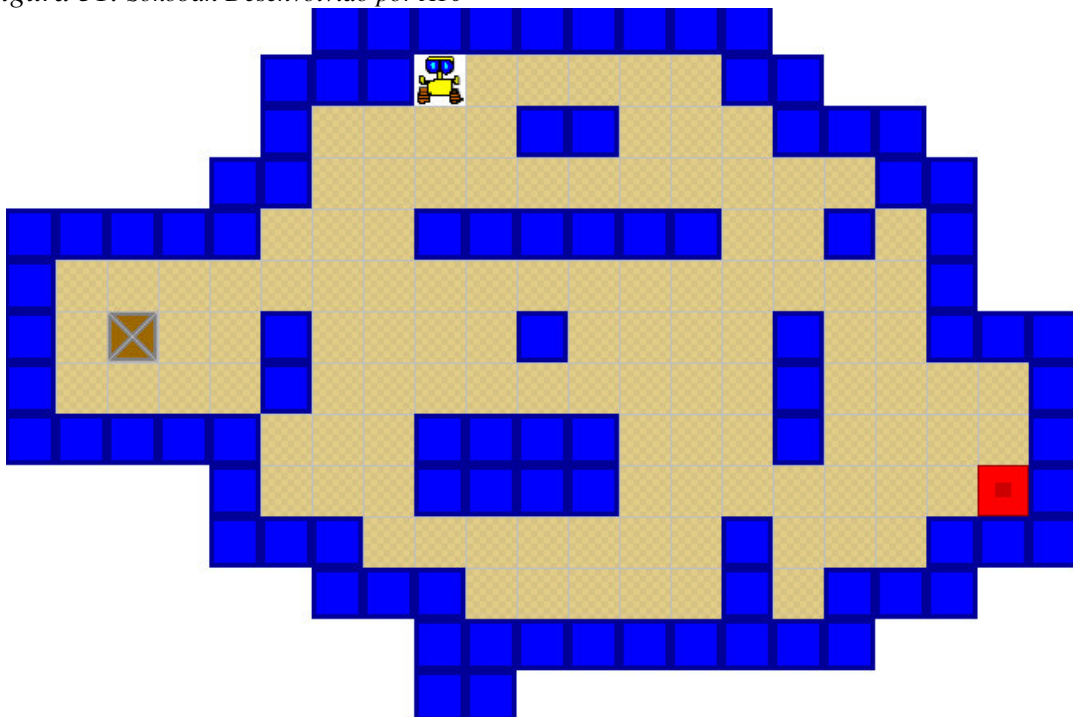
Na visão do designer, o usuário é alguém que gosta de ter tempo de pensar e poder voltar os obstáculos, pois o sapo pode andar para trás e voltar nos obstáculos. O jogo apresenta um nível mais difícil onde os caminhões, tartarugas e troncos são mais rápidos, aumentando as chances de cair na água ou ser atropelado, o que leva a pensar que o usuário gosta de níveis mais desafiadores.

COMPARAÇÃO:

O designer não se preocupou com a criação de nenhum signo metalinguístico, o que acaba por dificultar o entendimento e a jogabilidade do jogo, pois não têm como saber quais são os comandos do jogo e nem qual o objetivo. Os signos dinâmicos e estáticos apresentam uma ideia de um jogo simples e divertido. Apenas analisando os signos dinâmicos podemos ver um aumento de dificuldade na segunda fase do jogo, já que não existem signos metalinguísticos e os elementos usados na segunda fase são os mesmos da primeira.

A10 - SOKOBAN

Figura 31. Sokoban Desenvolvido por A10



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Instruções do jogo (*Move the pusher with the arrow keys. Push the crate to the destination*): Menciona os controles para mover o personagem principal e o objetivo do jogo
- Mensagem ao colocar a caixa no destino (*You win!*): Representa a vitória do jogador

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com início rápido, pois não apresenta nenhuma narrativa. No entanto, ele se preocupa com a apresentação dos comandos e do objetivo do jogo, o que facilita a jogabilidade do jogo.

SIGNOS ESTÁTICOS:

- Robô: Representa o personagem principal e controlável do jogo
- Quadrados azuis: Representam a parede ou limite de onde o personagem pode se locomover.
- Ladrilho: Representa a área onde o personagem pode se locomover livremente.
- Caixa: Objeto que deve ser levado ao objetivo
- Quadrado vermelho: Objetivo do jogo, onde a caixa deve ser levada.
- Área Branca: Representa o fundo do jogo.

Metamensagem:

O designer se baseia em um usuário que gosta do tema do Wall-E e de jogos que não apresentam uma interface detalhada, pois o jogo é feito apenas de formas quadrangulares, o que pode deixar o jogo repetitivo visualmente.

SIGNOS DINÂMICOS:

- Robô: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Caixa: O movimento das caixas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.

Metamensagem:

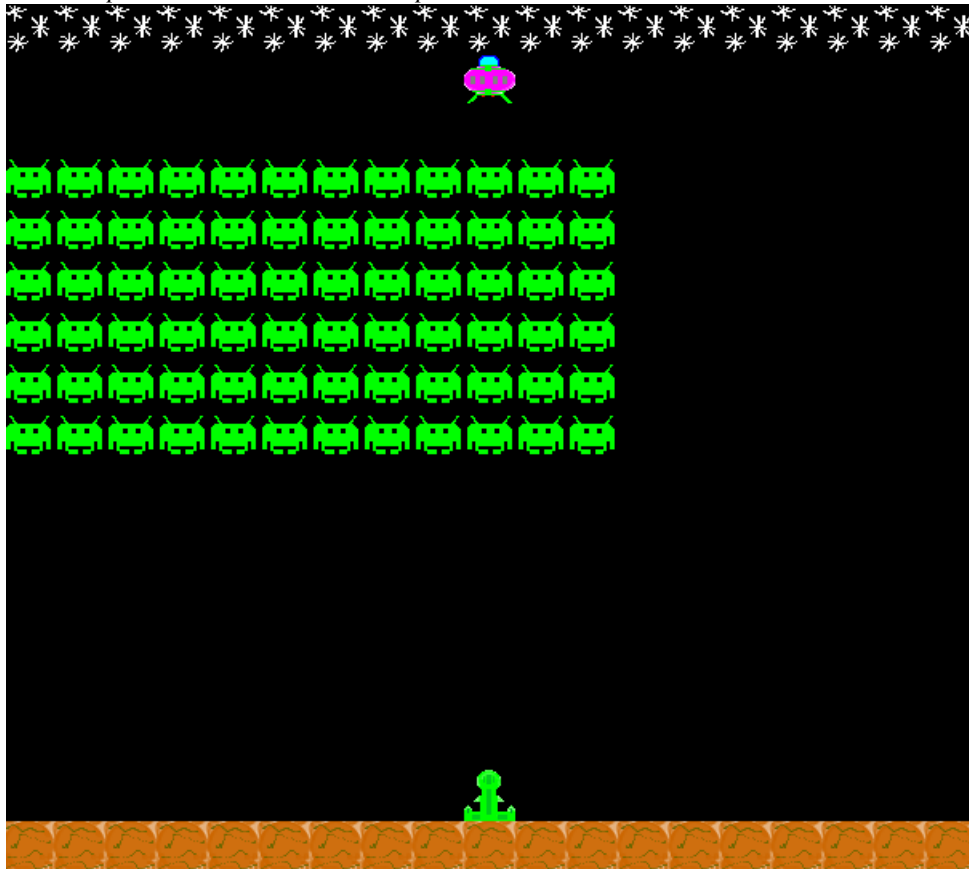
O designer baseia o jogo em um usuário que gosta de jogos com objetivos simples e com poucas interações, mas que necessitem de uma estratégia básica para poder vencer.

COMPARAÇÃO:

Durante a comparação, foi observado que os signos metalinguísticos não apresentam muitas informações sobre o jogo, o que pode dificultar o entendimento sobre ele. No entanto, em conjunto, todos os signos passam a ideia de um jogo simples e rápido.

A10 - SPACE INVADERS

Figura 32. Space Invaders Desenvolvido por A10



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS:

- Instruções do jogo (*Use the arrow keys to move the cannon. Press space to fire lasers at the aliens and mothership. Object of the game is to kill the mothership and avoid the bombs.*): Menciona os controles para mover e atirar com o canhão e os objetivos do jogo.
- Mensagem ao ser atingido por um míssil (*Your cannon was destroyed! Game over!*): Representa a derrota do jogador.
- Mensagem ao pouso do alien (*Alien invaded! Game over!*): Representa derrota do jogador.
- Mensagem ao destruir a nave mãe (*You win*): representa vitória do jogador e finalização do jogo.

Metamensagem:

Na visão do designer, o usuário é alguém que gosta de jogos com início rápido, pois não apresenta nenhuma narrativa do jogo. No entanto, ele se preocupa com a apresentação dos comandos e do objetivo do jogo, o que facilita o entendimento e a jogabilidade. O jogo apresenta diversas mensagens de feedback para determinadas ações do usuário.

SIGNOS ESTÁTICOS:

- Nave: Representa o personagem principal e controlável do jogo.
- Pedras: Representa o chão onde o canhão pode se locomover livremente.
- Alienígenas: Seres que representam perigo para o canhão.
- Nave lilás: Representa a nave mãe e objetivo do jogo.
- Estrelas: Representa o céu e o limite superior do jogo.

Metamensagem:

O jogo se baseia em elementos que representam alienígenas e objetos espaciais, o que está de acordo com um contexto de invasão ao planeta, ou seja, o designer vê o usuário como alguém que goste do conceito espacial. Os elementos são bastante coloridos, o que passa a ideia de um jogo divertido e não sério.

SIGNOS DINÂMICOS:

- Nave: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Alienígenas: O movimento dos alienígenas representa um aumento na dificuldade para destruí-los.
- Nave lilás: O movimento da nave lilás representa um aumento na dificuldade para destruí-la.
- Mísseis: O movimento dos mísseis representa ação causal que pode destruir o canhão.
- Laser: O movimento do laser representa ação causal para destruir as naves inimigas

Metamensagem:

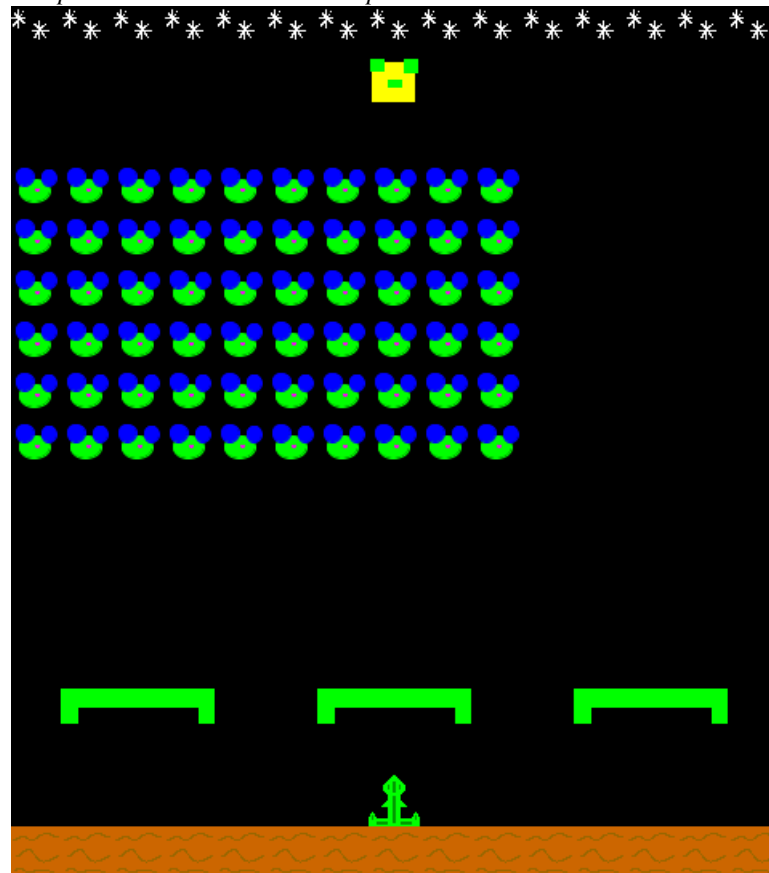
O designer utiliza diversos elementos que estão de acordo com a vida real, como mísseis e lasers como objetos de destruição, o que dá uma ideia de que o usuário gosta de jogos em que o objetivo é a destruição. Além disso, ele exige um pouco de rapidez por parte do usuário, pois o personagem principal já começa em uma área de perigo, além de apresentar diversos obstáculos que são perigosos ao personagem principal, o que torna o jogo mais difícil.

COMPARAÇÃO:

Durante a comparação foi observado que todos os signos estão de acordo, pois eles apresentam um contexto espacial, simplicidade e facilidade no jogo. No entanto, apenas nos signos dinâmicos podemos observar que o usuário precisa pensar rápido para se defender, pois o personagem principal já inicia em uma área de perigo.

A11 – SPACE INVADERS

Figura 33. Space Invaders Desenvolvido por A11



Fonte: PoliFacets

SIGNOS METALINGUÍSTICOS

- Descrição do jogo (*In this game the Laser Cannon has to kill the aliens before they kill him.*): Descreve o objetivo, quem é o personagem principal, os inimigos e a causa de derrota no jogo.
- Instruções do jogo (*Click left to move left. Click right to move right. Tap "space" to fire*): Informa quais os comandos necessários para jogar
- Mensagem ao ser atingido por míssil inimigo (*Your cannon was destroyed! Game over!*): Informa que o personagem principal foi destruído e o fim do jogo
- Mensagem quando nave inimiga pousa (*The alien fleet has landed! Game over!*): Informa a derrota do personagem principal e o fim do jogo

Metamensagem

Na visão do designer, o usuário é alguém que gosta de desafios e deve estar ciente dos perigos que irá encontrar. Além disso, ele passa a ideia de que o usuário gosta de receber feedbacks quando determinadas ações acontecem no jogo.

SIGNOS ESTÁTICOS

- Canhão: Personagem principal e controlável do jogo.
- Quadrado amarelo com verde: Nave mãe e objetivo principal do jogo
- Bola verde com azul: Nave inimiga que protege a nave mãe
- Estrela: Objeto decorativo
- Terra: Área em que o canhão pode se mover livremente

- Linhas Verdes: Objeto que empata que os tiros passem

Metamensagem

O jogo se baseia em elementos que representam alienígenas e objetos espaciais, o que está de acordo com um contexto de invasão alienígena, ou seja, o designer vê o usuário como alguém que goste do conceito espacial. Os elementos são bastante coloridos, o que passa a ideia de um jogo divertido e não sério.

SIGNOS DINÂMICOS

- Canhão: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Quadrado amarelo com verde: Seu movimento representa um aumento na dificuldade para destruí-lo.
- Bola verde com azul: Seu movimento representa um aumento na dificuldade para destruí-la.
- Mísseis: O movimento dos mísseis representa ação causal que pode destruir o canhão
- Laser: O movimento do laser representa ação causal para destruir as naves inimigas
- Linhas verdes: A mudança das linhas verdes representa a diminuição de sua vida

Metamensagem

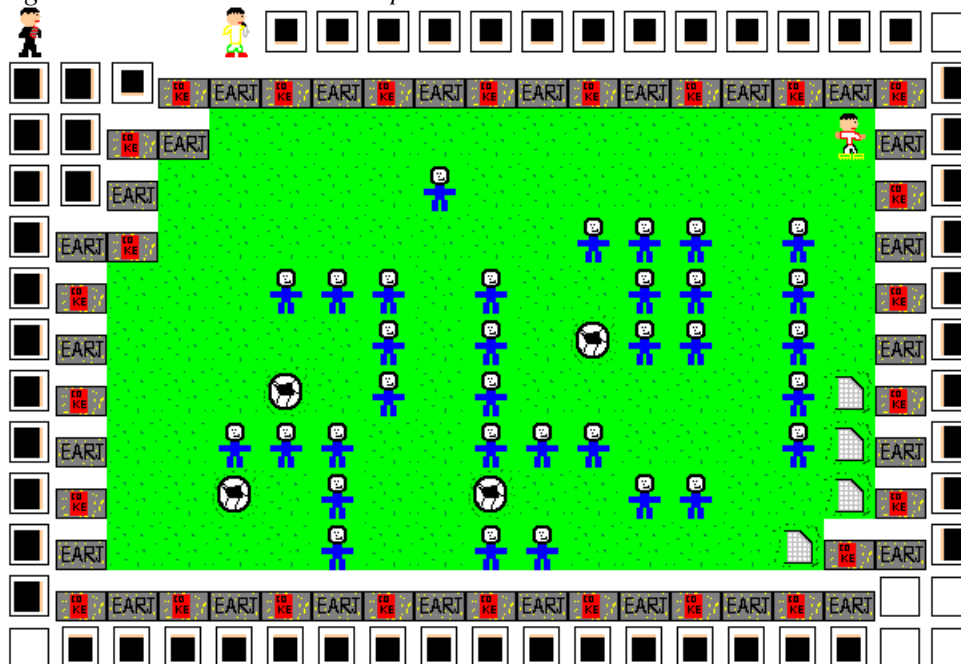
Na visão do designer, o usuário é alguém que considera que a melhor defesa é o ataque e a melhor maneira de proteger a Terra é destruindo seus inimigos. Mas, mesmo com este pensamento, ele utiliza de barreiras para se proteger enquanto ataca.

COMPARAÇÃO

Todos os 3 tipos de signos estão coerentes, passando a ideia de um jogo que exige uma estratégia de defesa e ataque. Os signos metalinguísticos deixam claro o objetivo do jogo, o significado dos objetos e os signos estáticos e dinâmicos reforçam essas informações.

A12 – SOKOBAN

Figura 34. Sokoban Desenvolvido por A12



Fonte: PoliFacets

SIGNOS METALINGUISTICOS

- Descrição do jogo (*This game is same as Sokoban but it is soccer version.*): Informa que o jogo é do estilo sokoban com uma temática diferente voltada para o futebol
- Intruções do jogo (*Box is soccer ball and pusher is soccer player. When ball goes to goal post, coach will move to judge. When all balls go to goalpost, coach will go on judge and you will go to level 2.*): Relaciona os objetos do jogo com os objetos originais do Sokoban, informa que a cada bola colocada no local certo o treinador se aproximará do juiz para ir para a próxima fase
- Mensagem ao finalizar o jogo (*You win!!*): Informa que o jogador conseguiu completar o jogo.

Metamensagem

Na visão do designer, o usuário é alguém que gosta do jogo Sokoban e de futebol e por isso apresenta um jogo usando as 2 temáticas. Além disso, o designer informa o objetivo dos elementos do jogo, e o objetivo do jogo, o que facilita a jogabilidade e entendimento inicial do jogo. O designer também muda o objetivo principal do jogo, que deixa de ser colocar as “caixas” nos “buracos” para fazer com que o treinador chegue no juiz.

SIGNOS ESTÁTICOS

- Jogador Branco: Personagem principal e controlável do jogo
- Jogadores azuis: Obstáculos que dificultarão o objetivo do jogo
- Arquibancada: Decoração de cenário
- Quadrados contendo propagandas: Decoração de cenário
- Bola: Objeto que deve ser colocado dentro da trave
- Trave: Objeto onde devem ser colocadas as bolas para cumprir o objetivo
- Personagem de preto: Decoração de cenário
- Personagem de branco: Decoração de cenário
- Grama: Área em que o jogador pode se movimentar livremente

Metamensagem

Na visão do designer, o usuário é alguém que gosta da temática de futebol. Além disso, o cenário apresenta elementos coloridos e engraçados e um time vestido com a blusa de outra cor, o que passa a ideia de um jogo divertido e ideal para passar o tempo, pois o jogo apresenta níveis curtos, e com inimigos, mesmo que apenas representativos, pois eles só servem como obstáculos.

SIGNOS DINÂMICOS

- Jogador Branco: Com a ação de movimento do jogador, é o personagem principal e controlado do jogo.
- Bola: O movimento das bolas é uma ação causada pelo jogador ao empurrá-las com o personagem principal.
- Personagem Branco: A movimentação do personagem branco em direção ao personagem de preto representa a quantidade de bolas que foram colocadas nas traves.

Metamensagem

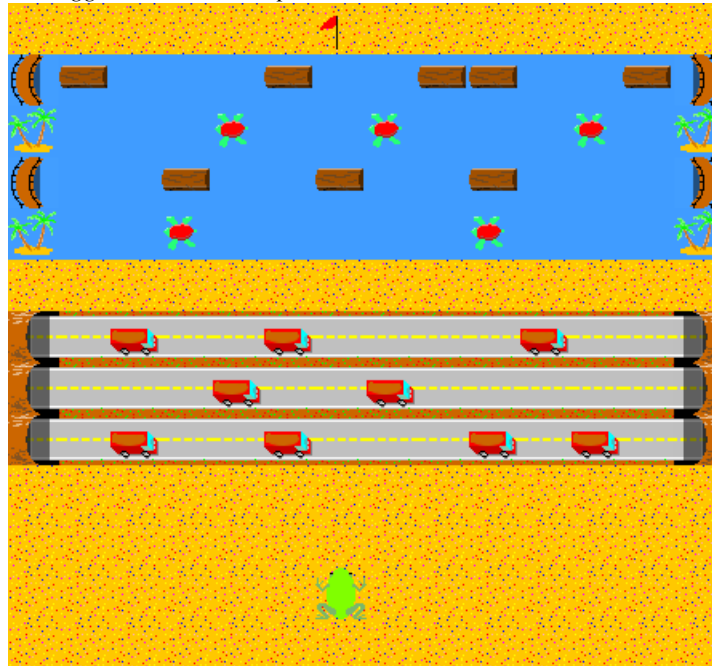
Na visão do designer, o usuário é alguém que gosta da temática de futebol onde o objetivo é colocar a bola dentro da trave. Além disso, o jogo faz com que o usuário tenha que pensar em um jeito de colocar a bola na trave, pois existem vários obstáculos que fazem com que a bola trave e não seja mais possível movê-la, passando a ideia de um jogo que precisa de raciocínio lógico.

COMPARAÇÃO

A combinação dos signos está bastante precisa, com exceção do juiz e do treinador, em que só é fácil de saber seu significado por causa dos signos metalinguísticos. Nos signos estáticos não dá para saber que eles irão se mover, pois estão fora do campo de futebol. Porém, isso não quer dizer que eles foram divergentes, pois no que os estáticos não comunicaram, os dinâmicos e metalinguísticos complementaram. Além disso, nenhum dos signos informou quais eram os controles necessário para se jogar, o que pode dificultar a jogabilidade do jogo, pois os controles não seguem o padrão dos jogos comuns.

A13 – FROGGER

Figura 35. Frogger Desenvolvido por A13



Fonte: PoliFacets

SIGNOS METALINGUISTICOS

- Descrição do jogo (*An old game re invented or re created just for you. Frogger is the frog who tries to survive in the street for getting into the goal. Enjoy*): Informa apenas em que jogo este foi baseado e um objetivo secundário do jogo
- Instruções do jogo (*you use arrows to move.*): Informa os controles necessários para jogar
- Mensagem ao tentar passar pelos túneis (*Cannot walk over tunnels*): Informa que o personagem não pode andar sobre os objetos laterais
- Mensagem ao chegar na bandeira (*I made it*): Informa que o jogador conseguiu cumprir o objetivo do jogo
- Mensagem ao cair na água (*I cannot swim*): Informa que o jogador não pode se mover sobre a água

- Mensagem ao tentar passar pela ponte (*Cannot walk over log makers. That's cheating!*):
Informa que tentar passar pela ponte é trapaça

Metamensagem

Na visão do designer, o usuário é alguém que já conhece o jogo Frogger mas que gostaria de jogar um remake do jogo. O designer mostra os controles do jogo e dá mensagens de feedback para determinadas ações que podem ocorrer no jogo, o que aumenta a jogabilidade e facilita o entendimento inicial do jogo.

SIGNOS ESTÁTICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idêntica ao do Frogger Original da seção 5.1.1.1

SIGNOS DINÂMICOS:

Idênticos aos do Frogger Original da seção 5.1.1.1

Metamensagem:

Idêntica ao do Frogger Original da seção 5.1.1.1

COMPARAÇÃO:

Nos três tipos de signos os elementos do jogo são apresentados com o mesmo sentido, sem nenhuma divergência. O designer também faz uso de signos estáticos e dinâmicos que facilitam o entendimento do jogo, já que ele usa elementos vistos no dia-a-dia apenas trocando o sentido dos objetos das bordas, como, por exemplo, a ponte, que não pode ser usada para se atravessar o rio.

APÊNDICE B – Análise dos testes dos alunos

Legenda:

P1: A descrição e as instruções são suficientes e bem explicadas?

P2: Este jogo contém algum bug?

P3: Quais e onde estão os bugs? Como você sugere consertá-los?

P4: Você tem alguma sugestão para melhorar o jogo? Qual sua sugestão?

Teste 1

Testador: A14

Desenvolvedor: A10

Jogo: *Frogger*

P1: *Yes*

P2: *Yes*

P3: *Level 1 does not have a bug but level 2 does you can't see the frog.*

P4: *Try to fix level 2*

Análise do aluno:

O testador relata que há um bug no nível 2, onde ele não consegue ver o sapo. Ele sugere que o nível 2 seja consertado.

Bugs encontrados:

1. Ao morrer, o jogo continua sendo executado.
2. A velocidade dos caminhões está muito baixa e não acompanha a velocidade em que os caminhões aparecem, então, com o passar do tempo, há muitos caminhões, impossibilitando a passagem do sapo.
3. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação

De acordo com o aluno, o nível 2 não está jogável pois não tem como ver o sapo, o que podemos sugerir que foi um problema com o tamanho do mapa de *tiles* deste nível, pois pude ver o sapo normalmente. No entanto, ele não relatou nenhum dos outros bugs, que podem ser vistos ambos no nível 1. O primeiro pode ser percebido quando, mesmo morrendo, o jogo continua sendo executado. O segundo pode ser visto apenas com o passar do tempo, pois no início não tem muitos caminhões para que se perceba o bug e o último bug já é bem complexo, pois é intuitivo que o testador não passe na frente do túnel, pois é de lá que o caminhão sai e se espera que a morte ali seja certa.

Teste 2

Testador: A14

Desenvolvedor: A10

Jogo: *Space invaders*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso o testador não conseguiu ver nenhum bug, assim como não fez nenhuma sugestão.

Bugs encontrados:

1. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação:

O jogador poderia ter encontrado o bug se tivesse esperado uma nave descer até chegar a uma linha acima do canhão, então ele teria percebido que era impossível destruir a nave inimiga. Mas como não é um bug que interfere no caminho feliz e que exige um tempo para ser verificado, o aluno não o encontrou.

Teste 3

Testador: A14

Desenvolvedor: A10

Jogo: *Sokoban*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

O testador afirma não ter encontrado nenhum bug no jogo. Assim como não fez nenhuma sugestão.

Bugs encontrados:

1. O jogo não possui limites nas extremidades do jogo, pois quando o personagem é movido além da parede, visivelmente não acontece nada, mas para movê-lo para o lado oposto é necessário que o jogador pressione a tecla para mover-se para o lado oposto à parede a quantidade de vezes +1 que ele moveu em direção a parede.

Comparação:

Para que o aluno conseguisse ver o bug encontrado, ele deveria fazer o inesperado, que era tentar entrar na parede, o que pode passar a ideia de que o aluno testou apenas o caminho feliz, que não possuía nenhum bug.

Teste 4

Testador: A02

Desenvolvedor: A05

Jogo: *Frogger*

P1: *There is no description, but the instructions are pretty clear.*

P2: *Yes*

P3: *- The game starts on the Level 2 worksheet.*

-- Fix by changing where it asks what worksheet to start on.

- When you reach the end of Level 1, it doesn't change to Level 2.

-- Change the frog's behavior so that when it sees the flag, it changes to worksheet 2.

P4: ---

Análise do aluno:

O testador relata que o jogo está iniciando no nível 2 e que para corrigir este problema, deve mudar a configuração que indica em qual mapa de *tiles* o jogo deve ser iniciado. Ele também relata que, ao finalizar o nível 1, o jogo não está indo para o nível 2 e para consertar este problema, o desenvolvedor deveria mudar o comportamento do sapo para que quando se encontrasse com a bandeira, mudasse para o nível 2.

Bugs Encontrados:

1. O jogo está iniciando no mapa de *tiles* do nível 2.
2. Ao finalizar o nível 1, o jogo para como se houvesse sido finalizado, quando deveria levar o sapo para o nível 2.
3. O sapo consegue passar por cima da ilha, do túnel e da ponte.
4. O sapo não consegue se mover para trás.
5. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

Neste caso, o testador encontrou 2 bugs que são vistos ao executar o jogo e seguir o caminho feliz. Se ela tivesse testados caminhos que deveriam ser impossíveis ela teria visto o terceiro bug. Além disso, como o jogo é sempre andando para cima/frente, ele pode não ter tido de usar o comando para baixo/atrás, não percebendo o bug 4. O quinto bug encontrado e não relatado se da pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho.

Teste 5

Testador: A02

Desenvolvedor: A05

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: - *The game starts on the Level 2 worksheet.*

-- *Change the settings so that it starts on the Level 1 worksheet.*

- *In Level 1, there is no final destination.*

-- *Just place the blue box somewhere on the screen.*

- *In Level 1, the box is in the corner, so the Sokoban can't push it anywhere.*

-- *Start the box somewhere else.*

- *In Level 2, the game resets in different spots each time.*

-- *I'm not sure if that's just my computer, or a glitch...*

P4: - *In Level 2, there are 3 crates and 2 blue spots.*

---*Try to only have 1 crate and 1 blue spot.*

Análise do aluno:

O testador inicia relatando que o jogo está iniciando do nível 2 e que o desenvolvedor deve mudar a configurações para que o jogo inicia a partir do mapa de *tiles* do nível 1. Ele relata também que, no nível 1, não existe destino final e que o desenvolvedor deve apenas colocar uma caixa azul (destino final) na tela. Além disso, neste mesmo nível, há uma caixa

que se encontra no canto da parede, então não pode ser movida para nenhum local e pede para que a caixa seja colocada em algum outro local. No nível 2, ele relata que toda vez que reinicia o jogo, ele está diferente, mas não sabe dizer se é um problema no jogo ou no próprio computador que ela está testando.

Na seção de sugestões, ele fala sobre o nível 2 possuir 3 caixas e 2 destinos finais e pede para que haja apenas 1 caixa e 1 destino final.

Bugs encontrados:

1. O jogo está iniciando no nível 2.
2. O primeiro nível já inicia finalizado, pois a única caixa deste nível já se encontra no destino final.
3. No nível 2, ao colocar uma das caixas em algum dos destinos finais, o jogo é finalizado.

Comparação:

Mesmo que aparentemente nós tenhamos encontrados os mesmos bugs, descrevemos, a não ser o primeiro, de maneiras diferentes. O que o testador descreve no segundo bug é que o jogo não tem destino final, quando, na realidade, a caixa já se encontra em cima do destino final, que se encontra no canto da parede do jogo e como eles têm o mesmo tamanho, impossibilitando que se veja o destino final a não ser que se arraste a caixa para outro local antes do jogo ser executado. Com relação ao quarto ponto citado pelo testador, deduzi que realmente fosse um problema na máquina dele, pois não consegui ver este problema. Com a sugestão dada pelo testador, o terceiro bug não existiria, então ela pode ter confundido um bug com uma sugestão, pois todas as caixas deveriam ser colocadas no objetivo para que o jogo finalizasse, e não apenas uma das caixas.

Teste 6

Tester: A02

Desenvolvedor: A10

Jogo: *Frogger*

P1: *No, there are no instructions at all, and there is no description either.*

P2: *Yes.*

P3: *1) When the frog dies, the game just keeps on playing and doesn't reset itself. To fix it, edit the behavior of the frog so that when it dies, the game resets.*

2) In level 2, you can't see where the frog starts, so that if you simply press the up arrow, you will most likely hit a truck and die. To fix it, change the size of the worksheets so that they are the same size.

3) When you win on Level 2, the game resets immediately, instead of stopping and telling the player they won. To fix this, make a totally new agent for the flag on Level 2, then change the behavior so that the game resets when you touch that flag and tells the player they won.

P4: *For improvement, try to not have so many trucks at the beginning of Level 2.*

Análise do aluno:

No primeiro ponto, o testador relata que o jogo continua sendo executado quando o sapo morre e não reinicia sozinho e que para corrigir este problema o desenvolvedor deve editar o comportamento do sapo para que o jogo reinicie quando ele morrer.

No segundo ponto, ele relata que, no nível 2, não é possível ver o sapo e ao se mover para cima o sapo o sapo simplesmente é atingido por um caminhão e morre. Ele pede para que o desenvolvedor corrija o tamanho do mapa de *tiles* para que elas tenham o mesmo tamanho.

Já no último ponto, o desenvolvedor fala que, ao vencer o nível 2, o jogo simplesmente reinicia, quando deveria mostrar uma mensagem de vitória e parar. Ele fala que, para corrigir este problema, o desenvolvedor deve criar um agente para a bandeira do nível 2 para que ele mostre a mensagem de vitória e pare o jogo quando o sapo tocá-la.

Na seção de sugestões, o testador pede que, como melhoria, que o nível 2 não tenha tantos caminhões.

Bugs encontrados:

1. Ao morrer, o jogo continua sendo executado.
2. A velocidade dos caminhões está muito baixa e não acompanha a velocidade em que os caminhões aparecem, então, com o passar do tempo, há muitos caminhões, impossibilitando a passagem do sapo.
3. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

Neste caso, o testador relatou os dois primeiros bugs. No entanto, o terceiro ponto citado pelo testador não é um bug, pois não prejudica e nem implica na jogabilidade do Frogger. Este ponto deveria aparecer como uma sugestão, pois a mensagem de vitória não precisa necessariamente aparecer para que o jogador saiba que ganhou. Já na seção de sugestões, ela fala que o jogo não precisa de tantos caminhões. No entanto, com o passar do tempo, os caminhões atrapalham totalmente a passagem do sapo, impossibilitando que o jogador tenha chance de passar, então esta sugestão deveria estar na seção de bugs, pois impossibilita que o jogador tenha alguma chance de vencer. O terceiro bug encontrado e não relatado se da pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou este caminho.

Teste 7

Tester: A02

Desenvolvedor: A10

Jogo: *Space Invaders*

P1: *There is no description, but the instructions are pretty clear.*

P2: *Yes*

P3: *When the player hits the mothership in Level 2, the game says "You Win" and then resets to Level 2. To fix this, make a new mothership to end the game. Then when the player wins, the level should switch back to Level 1.*

P4: *I don't have any suggestions.*

Análise do aluno:

O testador relata um problema onde o jogo, depois de finalizado no nível 2, reinicia para o nível 2 e cita que, para corrigir o problema, será necessário criar um novo agente da nave mãe para reiniciar a partir do nível 1.

Bugs encontrados:

1. O jogo está iniciando no nível 2.
2. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação:

Neste caso, o testador relatou um problema que realmente está ocorrendo, mas, para corrigi-lo, seria necessário outro método, pois o problema não é que o nível 2 está reiniciando para o nível 2 e sim que o jogo está iniciando a partir do nível 2, então, quando finalizado e reiniciado, ele volta para o nível 2. Então o correto seria corrigir a configuração que define em qual mapa de *tiles* o jogo deve iniciar. Além disso, ele não conseguiu perceber o segundo bug encontrado, pois seria necessário aguardar até que as naves inimigas descessem o suficiente para chegar a primeira linha acima do canhão.

Teste 8

Tester: A02

Desenvolvedor: A10

Jogo: *Sokoban*

P1: *There is no description, but the instructions are pretty clear.*

P2: *NO*

P3: *---*

P4: *Maybe make another level... You said you only needed one, but it would be more fun with 2.*

Análise:

Neste caso o testador afirma não ter encontrado nenhum bug no jogo. Mas sugere que o jogo tenha outro nível, pois, apesar do desenvolvedor ter dito que só precisaria de um, ter um segundo nível deixaria o jogo mais divertido.

Bugs encontrados:

1. O jogo não possui limites nas extremidades do jogo, pois quando o personagem é movido além da parede, visivelmente não acontece nada, mas para movê-lo para o lado oposto é necessário que o jogador pressione a tecla para mover-se para o lado oposto à parede a quantidade de vezes +1 que ele moveu em direção a parede.

Comparação:

Neste caso, o testador não encontrou nenhum bug, apesar do bug encontrado. Este não é um bug perceptível e não faz parte do caminho feliz, pois é necessário que o jogador tente “entrar na parede”, o que não faz muito sentido enquanto se joga um jogo com paredes.

Teste 9

Tester: A01

Desenvolvedor: A12

Jogo: *Sokoban*

P1: *not really t needs better description*

P2: *No*

P3: *none, its very well done*

P4: *I just recommend using better graphics*

Análise do aluno:

O testador não relatou nenhum bug, mas pediu para que o jogo tivesse uma descrição melhor. Ele também sugere que os gráficos do jogo sejam melhorados.

Bugs Encontrados:

1. Foi encontrado um bug visual do Sprite do jogador, pois, quando ele se move para a direita, a cor da roupa muda, mesmo vendo que quando ele desce o ângulo de visão da roupa do personagem é o mesmo e a roupa não muda de cor.
2. No nível 2, a partir da terceira bola colocada no objetivo, o contador para de funcionar e como o jogo só finaliza quando o contador termina, o jogo não é finalizado mesmo após cumprir todos os objetivos e continua sendo executado.

Comparação:

Nenhum dos bugs encontrados implica diretamente na jogabilidade, então é provável que o aluno não os tenha relatado porque conseguiu jogar e finalizar o jogo sem problemas que realmente atrapalhassem o jogador, pois o bug visual é apenas na mudança do Sprite e o segundo é o fato de o jogo não parar mesmo após finalizado.

Teste 10**Tester:** A01**Desenvolvedor:** A06**Jogo:** *Sokoban***P1:** *have more inthusiasm when writing description***P2:** *Yes***P3:** *screen is too big, the game cant fit in the given space to play the game***P4:** *I wouldn't do anything better its very good but could be tweeked***Análise do aluno:**

O testador relata que o mapa de *tiles* é grande demais e não cabe na tela, impossibilitando de ser jogado. Na seção de sugestões, ele cita que não faria nada melhor e está muito bom, mas poderia ser mexido um pouco.

Bugs encontrados:

1. Quando o personagem se move arrastando a caixa, a animação continua no sentido anterior a ação, por exemplo, se antes ele se movia para a direita, o Sprite apontava para a direita e se ele mover uma caixa para cima, o Sprite vai continuar apontando para a direita, sendo que se ele não estivesse empurrando a caixa, ele apontaria para cima normalmente.
2. Ao posicionar a caixa no objetivo em ambos os níveis 1 e 2, o jogo apresenta um erro e trava, o que torna impossível a continuação do jogo do nível 1 para o nível 2.

Comparação:

O aluno relata um problema do jogo ser grande e não caber na tela, mas ele esquece de dizer que isso ocorre apenas no nível 2, que é um nível normalmente inalcançável por causa do segundo bug encontrado, que não é citado pelo aluno. Ele também não cita o problema do primeiro bug encontrado, tendo em vista que esse é apenas um bug visual e que não implica na jogabilidade, é provável que ele não tenha o considerado como bug.

Teste 11**Tester:** A03**Desenvolvedor:** A04**Jogo:** *Sokoban***P1:** *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso, o testador relata não ter encontrado nenhum bug, assim como não fez nenhuma sugestão.

Bugs encontrados:

1. Há um bug visual na tela, onde existe a imagem de um segundo personagem principal no canto da tela, o que pode vir a confundir o jogador.
2. Após finalizar o nível 2, o jogo continua sendo executado.

Comparação:

Ambos os bugs encontrados não implicam na jogabilidade, o que pode ter sido motivo para que, mesmo que o aluno tenha percebido, não tenha relatado, pois o primeiro é muito perceptível e não precisa que o aluno tenha feito nada no jogo para ver que há um segundo personagem e o segundo ocorre apenas depois que o jogo é completamente finalizado.

Teste 12

Tester: A03

Desenvolvedor: A04

Jogo: *Frogger*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso, o aluno não relatou nenhum bug, assim como não fez nenhuma sugestão para o jogo.

Bugs encontrados:

1. O jogo está iniciando pelo nível 2.
2. Ao finalizar o nível 1, o jogo não está passando para o nível 2, apenas reinicia o jogo a partir do nível 2.
3. Os caminhões azuis não estão sendo excluídos do mapa de *tiles* quando chegam ao túnel, ou seja, quando um caminhão azul chega no destino final ele apenas para e todos os outros caminhões atrás dele começam a gerar um congestionamento bloqueando toda a estrada e impossibilitando a passagem do sapo.
4. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos *tiles* em que os caminhões são gerados. Além disso, se ele ficar parado neste *tile*, não são gerados mais caminhões.

Comparação:

Os primeiros bugs não necessariamente implicam na jogabilidade, já que, utilizando o PoliFacets, o testador pode mudar o nível e talvez por esse motivo o aluno não os tenha visto como bugs. O terceiro bug encontrado é visto com um pouco de tempo de jogo, já que os caminhões não iniciam já perto do túnel e por isso o testador talvez não o tenha visto. O quarto bug encontrado e não relatado se dá pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho.

Teste 13

Tester: A03

Desenvolvedor: A04

Jogo: *Space invaders*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso o testador não relatou nenhum bug, assim como não fez nenhuma sugestão para o jogo.

Bugs encontrados:

1. O jogo apresenta um bug onde, no nível 2, as naves saem dos limites da tela e depois voltam, apresentando um bug visual no jogo.
2. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação:

Por não ser um bug que implica na jogabilidade e impede que o jogador finalize o jogo, talvez o aluno não tenha considerado um problema visual como bug, o que leva a pensar que a definição de bug não esteja claro para o aluno. O jogador poderia ter encontrado o segundo bug se tivesse esperado uma nave descer até chegar a uma linha acima do canhão, então ele teria percebido que era impossível destruir a nave inimiga. Mas como não é um bug que interfere no caminho feliz e que exige um tempo para ser verificado, o aluno não o encontrou.

Teste 14

Tester: A03

Desenvolvedor: A08

Jogo: *Frogger*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso o testador não relatou nenhum bug no jogo, assim como não fez nenhuma sugestão também.

Bugs encontrados:

1. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

O único bug encontrado e não relatado se da pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho.

Teste 15

Tester: A03

Desenvolvedor: A08

Jogo: *Sokoban*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso, o testador não encontrou nenhum bug, assim como não fez nenhuma sugestão para o jogo.

Bugs encontrados:

Nenhum bug foi encontrado durante a análise do jogo.

Comparação:

O aluno não encontrou nenhum bug, assim como não encontramos nenhum durante a análise do jogo.

Teste 16

Tester: ekajava@earj.com.br

Desenvolvedor: Lola

Jogo: *Space Invaders*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso, o testador não encontrou nenhum bug, assim como não fez nenhuma sugestão.

Bugs encontrados:

1. Quando as naves inimigas estão um tile acima do canhão, é impossível destruí-las, pois o tiro do canhão surge no segundo tile acima do canhão, ignorando as naves que estão abaixo do tile de origem do tiro.

Comparação:

O aluno não relatou o bug encontrado, o qual precisa de um pouco de tempo para ser detectado, por é necessário que as naves desçam até o tile acima do canhão.

Teste 17

Tester: A04

Desenvolvedor: A03

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: *only the second level*

P4: ---

Análise do aluno:

Neste caso, o testador apenas informou que há bugs no nível 2, mas não mencionou nada sobre o bug. O testador não fez nenhuma sugestão ao jogo.

Bugs encontrados:

1. O nível 1 já se inicia finalizado, pois a caixa já se encontra no destino final.
2. No nível 2 não há um personagem jogável.
3. O jogo está iniciando no nível 2.

Comparação:

Neste caso, não tinha como o aluno realmente informar detalhes do bug, pois o jogo é completamente impossível de se interagir de forma normal, pois o nível 1 já se inicia finalizado e o nível 2 não tem nada que o jogador possa fazer além de olhar para a tela.

Teste 18

Tester: A04

Desenvolvedor: A13

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: *the worksheets are too big, they need to be fixed*

P4:

Análise do aluno:

Neste caso, o testador afirma que os mapas de *tiles* são grandes demais e precisam ser corrigidos. Ele não fez nenhuma sugestão para o jogo.

Bugs encontrados:

1. Uma das caixas está posicionada de forma que não é possível movimenta-la, tornando impossível o cumprimento do objetivo do jogo.

Comparação:

Como não foi possível o aluno jogar para testar o jogo, não há dados suficientes para que seja feita uma comparação.

Teste 19

Tester: A04

Desenvolvedor: A08

Jogo: *Frogger*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso, o testador não encontrou nenhum bug no jogo, assim como não fez nenhuma sugestão também.

Bugs encontrados:

1. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

O único bug encontrado e não relatado se dá pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho.

Teste 20

Tester: A04

Desenvolvedor: A08

Jogo: *Sokoban*

P1: *Yes*

P2: *No*

P3: ---

P4: ---

Análise do aluno:

Neste caso o testador não relatou nenhum bug no jogo, assim como não fez nenhuma sugestão também.

Bugs encontrados:

1. Uma das caixas está posicionada de forma que não é possível movimenta-la, tornando impossível o cumprimento do objetivo do jogo.

Comparação:

Em outros testes deste mesmo jogo, outros testadores informaram que não era possível jogar pois os mapas de *tiles* estavam maiores que a tela. Apesar disso, este testador não informou nenhum bug, então talvez a definição do que era um bug, para ele, não englobava o problema com os mapas de *tiles*.

Teste 21

Tester: A04

Desenvolvedor: A13

Jogo: *Frogger*

P1: *Yes*

P2: *Yes*

P3: *the second's level worksheet is too big so it doesn't fit*

P4: ---

Análise do aluno:

Neste caso, o testador relata que o mapa de *tiles* do nível 2 está grande demais e não dá pra ver o nível todo. Ele não faz nenhuma sugestão para o jogo.

Bugs encontrados:

1. No nível 2 não estão sendo gerados os troncos e nem as tartarugas, impossibilitando a passagem do sapo pela água.

2. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhos são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhos.

Comparação:

O primeiro bug encontrado não poderia ser relatado pelo aluno, já que ele afirma que o mapa de *tiles* do nível 2 está grande demais para que se possa jogar. Já o segundo bug encontrado e não relatado se dá pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho.

Teste 22

Tester: A05

Desenvolvedor: A02

Jogo: *Frogger*

P1: *Yes, It is very self explanatory*

P2: *Yes*

P3: *When you complete level one it stays on level one, even though there is a level 2. Change it so that when you complete level one, you change to level 2*

P4: *Just do what I said above, but other than that it looks good and is very easy to understand.*

Análise do aluno:

O testador relata que há um bug no nível 1 o qual, ao finalizá-lo, o jogo reinicia para este mesmo nível, mesmo havendo um nível 2. Ele pede para que o desenvolvedor corrija este problema para quando finalizar o nível 1, o jogo mude para o nível 2. Na seção de sugestões ele reforça o pedido para corrigir o bug e dá um feedback positivo sobre o jogo.

Bugs encontrados:

1. O jogo está iniciando em um nível 1 não jogável, possuindo outro mapa de *tiles* com o nível 1.
2. O nível 1 jogável já está iniciando finalizado, pois o sapo já se encontra no objetivo final do nível.
3. O sapo está conseguindo passar através das ilhas, pontes e túneis.
4. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhos são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhos.

Comparação:

O bug relatado pelo aluno não foi encontrado durante a análise do jogo, pois no nível 1 o jogo já se inicia finalizado e mesmo assim passa para o nível 2. O que provavelmente aconteceu foi que, quando ela iniciou o nível 1 já finalizado, foi direto para o nível 2, e quando reinicia passa para o nível 1 novamente, que já se encontra finalizado e passa para o segundo nível instantaneamente. Além disso, o testador não relatou que o sapo está conseguindo passar pelos obstáculos livremente, o que passa a ideia que ele não testou outros caminhos além do feliz. Já o último bug encontrado e não relatado se dá pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho, reforçando a ideia de que foi testado apenas o caminho feliz.

Teste 23

Tester: A05

Desenvolvedor: A02

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: *The bug is that the crate is in a bad place and the only place you can move it is the corner, which gets you no where. You need to place the block somewhere else.(That is in level one) There are no bugs in level two.*

P4: *I would just fix the bug that I pointed out, other than that its fabu!*

Análise do aluno:

Neste caso, o testador informa que há um problema com a localização da caixa, pois ela só poderia ser movida para o canto da parede, o que impossibilita a vitória do jogador. Na seção de sugestões, ele apenas pede para que o bug informado por ele seja corrigido e faz um elogio ao jogo.

Bugs encontrados:

Não foi encontrado nenhum bug durante a análise deste jogo.

Comparação:

Apesar de o aluno informar um problema com a caixa, é sim possível de movê-la normalmente até o objetivo. Ele apenas não tentou uma forma diferente e isso está relacionado apenas com a dificuldade do jogo.

Teste 24

Tester: A05

Desenvolvedor: A02

Jogo: *Space Invaders*

P1: *Yes*

P2: *No*

P3: *---*

P4: *I thank that it is too hard! But that is just my opinion. But I do really like the format of the aliens in the second level!*

Análise do aluno:

Neste caso, o testador não informou nenhum bug, mas agradeceu pela dificuldade do jogo e fez um elogio aos gráficos.

Bugs encontrados:

1. As naves inimigas das linhas de cima estão atirando e, como não há espaço para o tiro passar, a imagem do tiro fica presa e depois some, gerando um pequeno bug visual, mas que não implica na jogabilidade.
2. Algumas naves estão passando do limite da tela quando se movimentam para o lado direito, causando um bug visual com a nave e o plane de fundo do jogo.
3. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação

Apesar do aluno não ter relatado nenhum bug, o jogo apresenta 2 bugs visuais que não foram relatados, passando a ideia de que o aluno não considera problemas visuais no jogo

como bugs. Além disso, o terceiro bug, que é visto após algum tempo de jogo, também não foi relatado.

Teste 25

Tester: A05

Desenvolvedor: A10

Jogo: *Frogger*

P1: *There are no instructions or description...tsk tsk tsk*

P2: *Yes*

P3: *In the second level the worksheet is too big so I cannot see the frog when I start. You can fix this by making the worksheet smaller*

P4: *I think that the level two is too hard and you can fix that by making the trucks not come out as fast.*

Análise do aluno:

O testador informa que, no segundo nível, o mapa de *tiles* é grande demais e ele não consegue ver o sapo quando inicia o jogo. Ele cita que, para corrigir este problema, o desenvolvedor deve diminuir o tamanho do mapa de *tiles*. Na seção de sugestões, ele fala sobre o nível 2 ter um nível de dificuldade muito alta e para corrigir isso, o desenvolvedor deveria fazer com que os caminhões não fossem gerados tão rápido.

Bugs encontrados:

1. Ao morrer, o jogo continua sendo executado.
2. A velocidade dos caminhões está muito baixa e não acompanha a velocidade em que os caminhões aparecem, então, com o passar do tempo, há muitos caminhões, impossibilitando a passagem do sapo.
3. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

Neste caso, o aluno fala sobre um problema com o tamanho do mapa de *tiles* do nível 2, mas não impede que ele teste este nível. Na seção de sugestões ele fala sobre a dificuldade do jogo, mas na verdade, com o tempo, o sapo fica impossibilitado de passar pela estrada, pois como a velocidade dos caminhões não está acompanhando a velocidade de geração deles, acaba gerando uma barreira de caminhões, o que implica na jogabilidade, tornando-se um bug que só daria para perceber se o aluno tivesse esperado um tempo enquanto o jogo era executado. Já o último bug encontrado e não relatado se dá pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho, passando a ideia de que foi testado apenas o caminho feliz.

Teste 26

Tester: A05

Desenvolvedor: A10

Jogo: *Space Invaders*

P1: *There is no description, but the instructions are very nice.*

P2: *No*

P3: ---

P4: *It is very hard to tell the difference between level one and two, please make level one easier (maybe one or two less rows of aliens in level one) but level two is hard enough!*

Análise do aluno:

O testador não informou nenhum bug, mas fez uma sugestão sobre a similaridade de dificuldade entre os 2 níveis e pede para que torne o nível 1 mais fácil, pois o 2 já está difícil suficiente.

Bugs encontrados:

1. O jogo inicia a partir do nível 2
2. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação:

Apesar o aluno não ter relatado nenhum bug, ele fez uma ótima sugestão, já que os dois níveis estão realmente parecidos e não passam uma ideia de progressão no jogo. Apesar disso, o aluno não relatou que o jogo está iniciando no nível 2, o que é facilmente visto mas pode ser burlado pelas opções do PoliFacets. Além disso, para encontrar o segundo bug, era necessário que o aluno deixasse o jogo executando por um tempo apenas desviando dos tiros inimigos, para ver que era impossível destruir as naves quando elas estavam diretamente acima do canhão.

Teste 27

Tester: A05

Desenvolvedor: A10

Jogo: *Sokoban*

P1: *There is no description but I like the instructions.*

P2: *No*

P3: ---

P4: *I would put another level and make the background of the pusher thing match the floor because ya know...gotta look fabu*

Análise do aluno:

Neste caso, o testador não informou nenhum bug, mas fez a sugestão de colocar outro nível no jogo e melhorar a aparência do personagem do plano de fundo do jogo para parecer melhor.

Bugs encontrados:

1. O jogo não possui limites nas extremidades do jogo, pois quando o personagem é movido além da parede, visivelmente não acontece nada, mas para movê-lo para o lado oposto é necessário que o jogador pressione a tecla para mover-se para o lado oposto à parede a quantidade de vezes +1 que ele moveu em direção a parede.

Comparação:

Para que o aluno conseguisse ver o bug encontrado, ele deveria fazer o inesperado, que era tentar entrar na parede, o que pode passar a ideia de que o aluno testou apenas o caminho feliz, que não possuía nenhum bug.

Teste 28

Tester: A07

Desenvolvedor: *hey,*

Jogo: *i forgot*

P1: *to add something*

P2: *in the one i sent before*

P3: *yea, youy start on the 2nd level*

P4: *fix it*

Análise do aluno:

Não há dados suficientes para que seja feita uma análise, pois ele apenas informa que falta adicionar algo na descrição e que o jogo está iniciando no nível 2, sugerindo apenas que ele conserte isso.

Bugs encontrados:

Não houve possibilidade de testar, pois o aluno não informou de quem é o jogo e nem qual é o jogo.

Comparação:

Não há informações suficientes para que seja feita uma comparação.

Teste 29

Tester: A07

Desenvolvedor: A09

Jogo: *sokoban*

P1: *yea, i guess*

P2: *yes*

P3: *when you move the box, somtimes the dude doesnt move with it, and other times, the box moves 2 spaces*

P4: *yea, fix it*

Análise do aluno:

Neste caso, o testador relata que, algumas vezes, quando o personagem move a caixa, o personagem não se move, apenas a caixa e algumas vezes a caixa move 2 espaços. Na seção de sugestões, ele pede que o desenvolvedor corrija os bugs.

Bugs encontrados:

1. Os níveis 1 e 2 estão com problemas no tamanho do mapa de *tiles* e apenas o nível 3 é jogável.
2. Algumas vezes, ao empurrar a caixa, o personagem principal não se move, apenas a caixa.
3. Às vezes, apenas ao se aproximar da caixa, ela é empurrada.
4. O jogo é finalizado quando a caixa é posta um tile acima do objetivo.

Comparação:

Neste caso, o aluno não relatou o problema sobre os mapa de *tiles*, o que passa a ideia de que ele não vê problema do mapa de *tiles* como um bug. No entanto, ele relatou os problemas ao empurrar as caixas, mas a caixa não se move 2 espaços. Quando o segundo bug ocorre e você tenta se aproximar da caixa novamente, o terceiro bug ocorre, dando a ideia de que a caixa se moveu duas vezes apenas por ter empurrado uma vez. O aluno também não relatou o bug sobre o objetivo final, o que passa a ideia que ele pode ter deduzido que a caixa realmente era para ser posta apenas acima do objetivo e não em cima do objetivo final.

Teste 30

Tester: A08

Desenvolvedor: A01

Jogo: *sokobanworking*

P1: *no instructions*

P2: *Yes*

P3: *screen is too small and the person doesn't move once you hit play.*

P4: ---

Análise do aluno:

Neste caso, o testador relata um problema com as dimensões dos mapas de *tiles* do jogo e que o personagem principal não se move quando você inicia o jogo.

Bugs encontrados:

1. No nível 1 não existe personagem jogável
2. O mapa de *tiles* do nível 2 está cortado e não é possível encontrar o objetivo final

Comparação:

O aluno relata que não consegue mover o personagem, o que foi visto no nível 1 durante a análise do jogo, mas isso ocorre porque não há um personagem para ser movido. Já o problema que ele cita sobre o mapa de *tiles* foi visto no nível 2 do jogo, pois ele se encontra cortado. O aluno relatou ambos os bugs que foram encontrados, no entanto, faltou um pouco de detalhamento para que facilitasse encontrar estes bugs.

Teste 31

Tester: A08

Desenvolvedor: A01

Jogo: *Frogger*

P1: *no instructions*

P2: *Yes*

P3: ---

P4: ---

Análise do aluno:

Neste caso, o testador informou que há bugs, mas não deu nenhuma informação sobre eles.

Bugs encontrados:

1. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

Neste caso, o aluno informa que há bug no jogo, mas não fala nada sobre o que ou onde então os bugs. Ele pode, ou não, ter encontrado o bug listado, mas não há como confirmar, já que ele não ofereceu maiores informações sobre os bugs.

Teste 32

Tester: A08

Desenvolvedor: A04

Jogo: *Frogger*

P1: *yes*

P2: *yes*

P3: *in the second level, you can't see all of her screen. It blocks it.*

P4: *Make the play thing smaller*

Análise do aluno:

O testador relata que há problemas com as dimensões do mapa de *tiles* do segundo nível do jogo e que ele não consegue ver o mapa de *tiles* inteiro. Na seção de sugestões, ele pede para que o desenvolvedor diminua o tamanho do mapa de *tiles*.

Bugs encontrados:

1. O jogo está iniciando pelo nível 2.
2. Ao finalizar o nível 1, o jogo não está passando para o nível 2, apenas reinicia o jogo a partir do nível 2.
3. Os caminhões azuis não estão sendo excluídos do mapa de *tiles* quando chegam ao túnel, ou seja, quando um caminhão azul chega no destino final ele apenas para e todos os outros caminhões atrás dele começam a gerar um congestionamento, bloqueando toda a estrada e impossibilitando a passagem do sapo.
4. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

Apesar do aluno informar que não conseguiu jogar no nível 2 por causa do tamanho do mapa de *tiles*, todos os bugs encontrados podem ser vistos no nível 1. Com relação ao primeiro e segundo bugs encontrados, o aluno pode não ver como bug já que o PoliFacets oferece a opção de escolher o nível jogado. O terceiro bug pode ser visto esperando algum dos caminhões azuis chegar no final da pista, percebendo que ele fica parado, enquanto os vermelhos são excluídos dos mapas de *tiles* quando chegam ao final da pista. Já o último bug encontrado e não relatado se da pela ideia de ser intuitivo que o aluno não passe pela frente do túnel, pois é de onde o caminhão é gerado e provavelmente por este motivo o aluno não testou esse caminho, dando a ideia de que foi testado apenas o caminho feliz.

Teste 33

Tester: A08

Desenvolvedor: A04

Jogo: *Sokoban*

P1: *Yes*

P2: *No*

P3: *---*

P4: *---*

Análise do aluno:

Neste caso, o testador não relata nenhum bug, assim como não da nenhuma sugestão para o jogo.

Bugs encontrados:

1. O jogo apresenta um bug visual onde há um clone do personagem principal no canto da tela, mas que não é jogável.
2. Ao finalizar o segundo nível, o jogo continua sendo executado.

Comparação:

Ambos os bugs encontrados não implicam na jogabilidade, o que pode ter sido motivo para que, mesmo que o aluno tenha percebido, não tenha relatado, pois o primeiro é muito perceptível e não precisa que o aluno tenha feito nada no jogo para ver que há um segundo personagem e o segundo bug ocorre apenas depois que o jogo é completamente finalizado.

Teste 34**Tester:** A08**Desenvolvedor:** A03**Jogo:** *Sokoban***P1:** *Yes***P2:** *Yes***P3:** *the second level***P4:** ---**Análise do aluno:**

Neste caso, o testador apenas informou que há bugs no nível 2, mas não mencionou nada sobre o bug. O testador não fez nenhuma sugestão ao jogo.

Bugs encontrados:

1. O nível 1 já se inicia finalizado, pois a caixa já se encontra no destino final.
2. No nível 2 não há um personagem jogável.
3. O jogo está iniciando no nível 2.

Comparação:

Neste caso, não tinha como o aluno realmente informar detalhes do bug, pois o jogo é completamente impossível de se interagir de forma normal, pois o nível 1 já se inicia finalizado e o nível 2 não tem nada que o jogador possa fazer além de olhar para a tela.

Teste 35**Tester:** A08**Desenvolvedor:** A03**Jogo:** *frogger***P1:** *Yes***P2:** *Yes***P3:** *i think is good***P4:****Análise do aluno:**

Neste caso, o testador relata que há bugs no jogo, porém não cita o que ou onde estão os bugs, dando apenas um feedback positivo sobre o jogo, falando que acha que está bom.

Bugs encontrados:

Não foi encontrado nenhum bug, pois esse jogo não foi encontrado no repositório.

Comparação:

Não há como fazer uma comparação, visto que o jogo citado pelo aluno não foi encontrado.

Teste 36

Tester: A08

Desenvolvedor: A03

Jogo: *space invaders*

P1: *Yes*

P2: *Yes*

P3: *yes, in the second level*

P4: ---

Análise do aluno:

Neste caso, o testador relata que há bugs no jogo, mas não diz o que são os bugs, apenas que estão no nível 2.

Bugs encontrados:

1. Todos os níveis estão com bugs no tamanho do mapa de *tiles*.

Comparação:

O aluno fala que há bugs no nível 2, mas não cita nada sobre eles. Como todos os níveis estão com problemas nos mapas de *tiles*, deduzi que o bug que o aluno fala que existe, está relacionado com o tamanho dos mapas de *tiles*, inclusive do nível 2.

Teste 37

Tester: A08

Desenvolvedor: A13

Jogo: *Frogger*

P1: *no instructions*

P2: *Yes*

P3: *the probability of the turtles and trunks coming out was not very high and you died in the water.*

P4: *Increase the possibility of turtles and trunk coming out*

Análise do aluno:

O testador relata que há um bug onde os troncos e tartarugas não têm muitas chances de aparecerem, impossibilitando que o jogador possa atravessar. Na seção de sugestões, o testador pede para que as chances de surgirem troncos e tartarugas sejam aumentadas.

Bugs encontrados:

1. O jogo está iniciando no nível 2.
2. No nível 2, o mapa de *tiles* está muito grande, cortando grande parte do local que o sapo se encontra no início do jogo.
3. Há um bug visual no cenário do nível 2, onde há um pedaço de água fora dos limites do mapa de *tiles*.
4. Há vários níveis no jogo que já iniciam finalizados e outros que não contem nada.
5. No nível 2 não é possível passar pelo primeiro rio, pois não são gerados troncos ou tartarugas.

Comparação:

O aluno fala sobre as chances dos troncos e tartarugas estarem baixar de aparecer, mas por mais que tenhamos esperado, nenhuma tartaruga ou tronco apareceu. O jogo possui muitos níveis e eles estão completamente desconectados, não sabendo qual deveria ser o primeiro nível, então poderia ter sido feito uma sugestão para que o desenvolvedor organizasse os níveis. O bug visual não foi relatado, passando a ideia de que o testador não considera um problema visual no jogo como um bug. O aluno também não relatou o bug do problema do mapa de *tiles*, já que ainda era possível ver apenas um pedaço do sapo, ela conseguiu jogar e provavelmente não o viu como um bug.

Teste 38**Tester:** A09**Desenvolvedor:** A07**Jogo:** *Space Invaders***P1:** *Yup***P2:** *No***P3:** ---**P4:** ---**Análise do aluno:**

O testador relata que não há nenhum bug, assim como não faz nenhum sugestão para o jogo.

Bugs encontrados:

1. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação:

O jogador poderia ter encontrado o bug se tivesse esperado uma nave descer até chegar a uma linha acima do canhão, então ele teria percebido que era impossível destruir a nave inimiga. Mas como não é um bug que interfere no caminho feliz e que exige um tempo para ser verificado, o aluno não o encontrou.

Teste 39**Tester:** A09**Desenvolvedor:** A07**Jogo:** *sokoban***P1:** *Yeah***P2:** *No***P3:** ---**P4:** ---**Análise do aluno:**

Neste caso, o testador relata que não há nenhum bug, assim como não faz nenhuma sugestão ao jogo.

Bugs encontrados:

Não foi encontrado nenhum bug durante a análise deste jogo.

Comparação:

Assim como o aluno relatou que não havia bugs, também não encontramos nenhum bug.

Teste 40

Tester: A10

Desenvolvedor: A02

Jogo: *Frogger*

P1: *Yes*

P2: *Yes*

P3: *When you press play it goes straight to Level 2 not Level 1*

The 1st time you get to the logs on level 2 it doesn't send enough out at once so you have to stay on the road and wait but by then you are dead

P4: *No I don't have any suggestions.*

Análise do aluno:

O testador relata que o jogo está sendo iniciado no nível 2 e, além disso, ele alega que no nível 2 não estão surgindo troncos suficientes para que o sapo atravessasse para o outro lado, tendo que esperar na estrada correndo risco de morrer.

Bugs encontrados:

1. O jogo está iniciando no nível 2
2. O jogo está iniciando em um nível 1 não jogável, possuindo outro mapa de *tiles* com o nível 1.
3. O nível 1 jogável já está iniciando finalizado, pois o sapo já se encontra no objetivo final do nível.
4. No nível 2 não estão aparecendo troncos e tartarugas no primeiro rio, impedindo que o sapo possa atravessar para o outro lado.
5. O sapo está conseguindo passar através das ilhas, pontes e túneis.
6. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos *tiles* em que os caminhões são gerados. Além disso, se ele ficar parado neste *tile*, não são gerados mais caminhões.

Comparação:

O aluno relata o primeiro bug encontrado e o quarto. No entanto, ele não fala sobre todos os mapas de *tiles* bugados, o que pode ser pelo motivo de ele ter levado em conta apenas aqueles que estavam funcionando normalmente. Ele também não cita o problema do nível 1 já estar sendo iniciado finalizado, então como ele não conseguiu testar este nível, levou em conta apenas o nível 2. Também não foi relatado o problema do sapo conseguiu passar pelos obstáculos livremente, mas como, no nível 2, que foi o único testado, não há como chegar nestes obstáculos, não podemos levar em conta ele não ter visto este problema, assim como o sexto bug.

Teste 41

Tester: A10

Desenvolvedor: A02

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: *Level 2 doesn't load*

Also the second time i tried opening the game Level 1 didn't load too.

P4: *No, overall it was a good game*

Análise do aluno:

O testador relata que há um bug no nível 2, impedindo que ele seja carregado pelo jogo. Além disso, quando ele tentou abrir o nível 1 pela segunda vez, ele também não foi carregado pelo jogo.

Bugs encontrados:

Não foi encontrado nenhum bug durante a análise deste jogo.

Comparação:

Neste caso, o aluno afirma não ter conseguido carregar nenhum dos níveis do jogo, mas na seção de sugestões dá um feedback positivo sobre o jogo. Outros alunos testaram este jogo e conseguiram analisá-lo e relatar os bugs normalmente, então é provável que tenha ocorrido algum problema no PoliFacets quando este aluno foi abrir o jogo ou algum outro problema desconhecido.

Teste 42

Tester: A10

Desenvolvedor: A02

Jogo: *Space Invaders*

P1: *Yes*

P2: *Yes*

P3: *Level 1 is fine but when it loads to level two the aliens are all over the place and some alien bullets haven't been destroyed also some of the back ground is sticking out.*

P4: *Fix level 2 and make the worksheets the same size so nothing is sticking out*

Análise do aluno:

O testador relata que está tudo bem no nível 1, mas, no nível 2, as naves inimigas estão saindo dos limites visuais do jogo e suas balas não estão sendo destruídas. Ele também informa um problema visual na tela de fundo quando as naves saem dos limites do jogo. O testador sugere que o nível 2 seja consertado e que os mapas de *tiles* sejam postos do mesmo tamanho para que não haja problemas.

Bugs encontrados:

1. As naves inimigas das linhas de cima estão atirando e, como não há espaço para o tiro passar, a imagem do tiro fica presa e depois some, gerando um pequeno bug visual, mas que não implica na jogabilidade.
2. Algumas naves estão passando do limite da tela quando se movimentam para o lado direito, causando um bug visual com a nave e o plano de fundo do jogo.
3. Naves inimigas que estão 1 quadrado acima do canhão não podem ser destruídas, pois o tiro do canhão está surgindo no segundo tile acima do canhão, ignorando o que está no tile entre o canhão e o tiro.

Comparação

Neste caso, o aluno não teve dificuldades de encontrar nenhum dos bugs visuais, pois ambos foram relatados por ele. No entanto, o terceiro bug, que precisa de um tempo para poder ser visto, não foi informado pelo aluno.

Teste 43

Tester: A10

Desenvolvedor: A05

Jogo: *Frogger*

P1: *Yes*

P2: *Yes*

P3: *A lot of logs spawned and the levels were mixed up.*

P4: *Make less logs spawn and Make level on start and make level 2 end it*

Análise do aluno:

Neste caso, o testador relata que há um bug na quantidade de troncos saindo e que os níveis estão trocados. Ele sugere que sejam gerados menos troncos e que o jogo inicie a partir do nível 1.

Bugs Encontrados:

1. O jogo está iniciando no mapa de *tiles* do nível 2.
2. Ao finalizar o nível 1, o jogo para como se houvesse sido finalizado, quando deveria levar o sapo para o nível 2.
3. O sapo consegue passar por cima da ilha, do túnel e da ponte.
4. O sapo não consegue se mover para trás.
5. Pelo sapo não morrer quando está acima do caminhão e apenas a frente dele, existe um bug onde o sapo pode passar livremente pelos tiles em que os caminhões são gerados. Além disso, se ele ficar parado neste tile, não são gerados mais caminhões.

Comparação:

Primeiramente o aluno relata um bug sobre a quantidade de troncos sendo gerados, o que não é um bug, apenas implica na dificuldade do jogo, então este bug, na verdade, deveria estar na seção de sugestões falando sobre a dificuldade do jogo. Dos bugs encontrados, o aluno relatou apenas o primeiro. O segundo bug, que faz parte do caminho feliz, não foi relatado, o que me leva a pensar que, quando ele finalizou o nível 1, ele trocou para o segundo nível usando o menu do PoliFacets. Ele também não relatou que o sapo estava conseguindo passar através dos obstáculos do jogo livremente, passando a ideia de que testou apenas o caminho feliz, mesmo não tendo relatado o problema do objetivo final do nível 1. Também não foi relatado o problema do sapo não conseguir se mover para trás e nem o dos tiles geradores dos caminhões, reforçando a ideia de que foi testado apenas o caminho feliz do jogo.

Teste 44

Tester: A10

Desenvolvedor: A05

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: *It doesn't let me win the game. Also it is impossible to get both destinations and move some boxes up.*

P4: *Move the box so you can move it up*

Análise do aluno:

O testador relata que não é possível vencer o jogo e que também é impossível chegar aos destinos e mover algumas caixas. Ele sugere que algumas caixas sejam mudadas de lugar para que seja possível movê-las.

Bugs encontrados:

1. O jogo está iniciando no nível 2.
2. O primeiro nível já inicia finalizado, pois a única caixa deste nível já se encontra no destino final.
3. No nível 2, ao colocar uma das caixas em algum dos destinos finais, o jogo é finalizado.

Comparação:

O primeiro problema que o aluno cita é que é impossível vencer o jogo, no entanto, o nível 1 já inicia vencido e o nível 2 só precisa que uma das caixas seja colocada no destino para vencer. Além disso, todas as caixas podem ser movidas em direção a pelo menos 1 dos objetivos finais, já que o outro realmente é impossível de se alcançar, mas não impede que o jogador vença.

Teste 45

Tester: A12

Desenvolvedor: A06

Jogo: *Sokoban*

P1: *Yes*

P2: *Yes*

P3: *I can't go level 2. When cheese go over the house, The game stopped.*

P4: *There are a lot of cheese but there is just one house. You need to match number of cheese and house.*

Análise do aluno:

O testador relata que não consegue ir para o nível 2 do jogo. Quando o queijo está na casa, o jogo para de funcionar. Na seção de sugestões, ele informa que têm muitos queijos e apenas uma casa, e o número de casas deviam ser correspondente ao número de queijos.

Bugs encontrados:

1. Quando o personagem se move arrastando a caixa, a animação continua no sentido anterior a ação, por exemplo, se antes ele se movia para a direita, o Sprite apontava para a direita e se ele mover uma caixa para cima, o Sprite vai continuar apontando para a direita, sendo que se ele não estivesse empurrando a caixa, ele apontaria para cima normalmente.
2. Ao posicionar a caixa no objetivo em ambos os níveis 1 e 2, o jogo apresenta um erro e trava, o que torna impossível a continuação do jogo do nível 1 para o nível 2.

Comparação:

O aluno relata um problema que não o deixa ir para o nível 2 e isso está relacionado com o segundo bug encontrado. No entanto, ele não relata o bug visual apresentado como primeiro bug encontrado, o que dá a ideia de que ou ele não percebeu o problema ou não considera um problema visual no jogo como bug. Sobre a sugestão, há vários estilos do jogo Sokoban, onde alguns precisam de um objetivo para cada caixa e outro não, então não é realmente necessário que haja um número de objetivos igual ao número de caixas.

Teste 46

Tester: A12

Desenvolvedor: A06

Jogo: *Frogger*

P1: *Yes*

P2: *Yes*

P3: *Frog can go out of the map so frog can go to flag without passing gmae.*

P4: *No*

Análise do aluno:

O testador relata que o sapo pode passar por fora do mapa, chegando ao destino sem passar pelos obstáculos.

Bugs encontrados:

1. O sapo pode passar livremente pela ponte, ilha e túnel.
2. Há bugs visuais mostrando objetos fora dos limites do jogo.
3. Há um problema com a destruição dos caminhões do cenário, fazendo com que o jogo dê uma pequena travada quando um caminhão é destruído e fazendo com que dê uma travada significativa quando dois ou mais caminhões são destruídos ao mesmo tempo.
4. Quando finalizado o segundo nível, o jogo mostrar uma mensagem de vitória com e, ao pressionar “OK”, a mensagem é fechada e reaberta para ser fechada novamente, num loop infinito.

Comparação:

O aluno informa o problema do sapo passando por cima dos obstáculos livremente, mas não fala nada sobre os outros bugs encontrados. Os objetos fora dos limites são bem perceptíveis mas não implicam na jogabilidade, então é provável que este seja o motivo de ele não ter relatado como bug. O terceiro bug só é visto com o passar do tempo, pois demora um pouco para que vários caminhões sejam destruídos ao mesmo tempo. Já o quarto bug, também muito perceptível e não citado pelo aluno, é incluso no caminho feliz, talvez não tenha sido citado porque ocorre depois da finalização do último nível do jogo e o aluno tenha simplesmente fechado a página após a conclusão.